Education Endowment Foundation

Reciprocal Reading effectiveness trial

Further appendices

February 2026

Neus Torres Blas, Emma Forsyth, Andrés Cueto, and Patrick Taylor

BIT

The Education Endowment Foundation (EEF) is an independent charity dedicated to breaking the link between family income and education achievement. We support schools, colleges, and early years settings to improve teaching and learning for 2–9-year-olds through better use of evidence.

We do this by:

- **Summarising evidence**. Reviewing the best available evidence on teaching and learning and presenting in an accessible way.
- **Finding new evidence.** Funding independent evaluations of programmes and approaches that aim to raise the attainment of children and young people from socio-economically disadvantaged backgrounds. Putting evidence to use.
- **Putting evidence to use**. Supporting education practitioners, as well as policymakers and other organisations, to use evidence in ways that improve teaching and learning.

We were set-up in 2011 by the Sutton Trust partnership with Impetus with a founding £125m grant from the Department for Education. In 2022, we were reendowed with an additional £137m from government, allowing us to continue our work until at least 2032.

For more information about the EEF or this report please contact:

The Education Endowment Foundation
5th Floor, Millbank Tower,
21–24 Millbank,
London,
SW1P 4QP

0207 802 1653

info@eefoundation.org.uk

www.educationendowmentfoundation.org.uk

Contents

Appendix E: FFT-School - memorandum of understanding

FFT Reciprocal Reading

Evaluation: Randomised Controlled Trial

MEMORANDUM OF UNDERSTANDING

About this document

We use online PandaDocs to make the process for ordering quick and simple for schools.

To confirm your place in the evaluation project you will need to:

- Read the Reciprocal Reading Randomised Controlled Trial Memorandum of Understanding in full to ensure participation in the project and responsibilities of the school are fully understood
- Complete the school information requested in the order form
- Confirm and sign the order form by clicking in the signature box

We will automatically be notified when you have completed the document so we can then process your order. If you need any assistance, please do not hesitate to get in touch and we will be happy to help.

Secure and Legally Binding

PandaDoc eSignatures are secure, legally binding and compliant with the UK Electronic Communications Act.

Any Browser or Mobile Device

This online PandaDoc is a cloud-based document which you can access and approve from all devices and browsers.

Evaluation: Randomised Controlled Trial

Thank you for your interest in the evaluation of Reciprocal Reading, delivered by FFT. The Education Endowment Foundation (EEF) have funded researchers at the Behavioural Insights Team (BIT) to conduct an independent evaluation of the programme.

This memorandum of understanding (MoU) explains what your school's participation in the project will entail. If you agree to take part, and accept the terms and conditions outlined in this MoU, please complete the questions at the end of this form and electronically sign. If you have any questions about the project and Reciprocal Reading before completing the MoU, please contact FFT (reciprocalreading@fft.org.uk).

Aims of the evaluation

The aims of this evaluation are to:

1. Evaluate the impact of FFT's Reciprocal Reading programme (Reciprocal Reading) on pupils' reading attainment and comprehension.
2. Understand factors that influence successful implementation of Reciprocal Reading.

This EEF project is part of a wider DfE funded programme called the 'Accelerator Fund' (AF). The AF aims to increase access to evidence-informed programmes.

The Project

Reciprocal Reading is designed to train teachers/teaching assistants (TA's) to work with groups of year 5 & 6 pupils, identified as having reduced comprehension skills, particularly those with a disparity between their ability to decode and to comprehend. Using a randomised controlled trial (RCT), the impact of Reciprocal Reading on reading attainment will be compared with a business as usual 'control group'. Schools will be randomly assigned to one of the following two groups for the duration of the project. These are:

1. ***Reciprocal Reading (Intervention group – 150 schools):*** Schools in the intervention group will deliver the Reciprocal Reading programme with selected pupils in Year 5 and Year 6 in 2023/24. 2. ***Business as usual (Control group – 150 schools):*** Schools in the control group will continue with usual teaching of comprehension, including any targeted programmes they currently use in Year 5 and 6 in 2023/24.

Eligibility Criteria

- Your school must not have used Reciprocal Reading previously or be using it.

- Your school must not be delivering the Inference Programme to pupils in your school.

Benefits of taking part

Reciprocal Reading (Intervention group)

Schools who are allocated to deliver Reciprocal Reading in academic year 2023/ 24 will receive a highly subsidised version of the programme for £215 +VAT (Approximately 7.5% of the normal cost of the programme at £2,800). This includes everything the school requires to deliver Reciprocal Reading as an intervention, including two days of face-to-face training for 3 school staff, book resources and programme manuals, two 1-1 online school support meetings from FFT and full support from FFT's literacy team. Schools can also receive the reading attainment results for their pupils at the end of the research project.

There is rigorous evidence on the effectiveness of Reciprocal Reading. A previous EEF efficacy trial showed that pupils who completed the targeted Reciprocal Reading intervention gained on average two additional months of progress in overall reading and reading comprehension, compared to equivalent pupils in the control group schools.

'Business as usual' (Control group)

Schools in the control group will receive £1,000 following the completion of all the evaluation requirements with the identified pupils in 2023/24. Schools can also receive the reading attainment results for their pupils at the end of the academic year. After the evaluation has finished, the 'control' schools may purchase at a reduced price the online training for Reciprocal Reading from FFT, for use from September 2024.

Random allocation is essential to the evaluation as it is the best way of investigating what effect Reciprocal Reading had on pupils' reading. It allows the research team to compare the progress made by pupils in each of the two groups. It is important that schools understand and consent to this process.

What does the Reciprocal Reading programme involve?

a. Schools will be asked to identify a teacher to coordinate the delivery of Reciprocal Reading in their school, a 'teacher coordinator'.
b. A teacher coordinator from the schools in the intervention group will attend a preliminary online briefing (60 minutes).
c. All year 4 & 5 pupils (academic year 2022/23) identified as being below the expected standard in reading comprehension will be assessed on their comprehension and decoding skills in May/June 2023 using FFT's screening tool. These assessments will be used to select a group of 12 pupils across both year groups who will also complete a standardised reading pre-test (GL Assessment New Group Reading Test) in June/July/September 2023. The tests will be administered in schools by BIT's Evaluation Team which will commission Qa Research to visit schools and administer tests directly to pupils. These school visits will follow a strict safeguarding policy requiring any project staff to undergo vetting before coming into contact with pupils. All pupils who complete a pre-test (as outlined in b.) will also complete a post-test after programme delivery is completed in the spring/summer term 2024. If selected pupils are absent on the testing day when Qa Research visits the schools, school staff may be

asked to administer the NGRT tests independently. School staff will be instructed on how to administer the tests by Qa Research Assessors during school visits.

d. All teachers and teaching assistants will complete a survey at the end of the project (summer term 2024).

e. ***All schools*** will be asked to identify 12 – 16 pupils across Years 5 and 6 (in academic year 2023/24) likely to benefit from Reciprocal Reading.

f. ***Schools in the control group*** will be asked to continue with their usual teaching with year 5 and 6 pupils in 2023/24.

g. ***Schools in the intervention group*** will be asked to meet the requirements outlined below:

- The school's school lead coordinator attends an initial briefing meeting.
- The school lead coordinator and two other teachers/TAs (or 3 members of staff from the school) that will deliver Reciprocal Reading attend 2 days of face-to-face training at a regional training venue and take part in two online support meetings (90 minutes each) with the FFT Reciprocal Reading team.
- Deliver Reciprocal Reading over 12-16 weeks, twice a week for 20 – 30 minutes. • Complete online registers of pupil attendance for the programme.
- Some schools will be asked to become case study schools and participate in classroom observations, staff interviews and focus groups with pupils.

Use of Data

What will happen to data collected as part of the study?

All pupil data will be processed in accordance with the General Data Protection Regulation (2018) and Data Protection Act (2018). Detailed information about how participant data will be used for this project can be found in FFT's privacy notice and BIT's Evaluation Team's privacy notice. These include information about the legitimate basis for processing data, and all relevant information about how school staff and pupils' data will be collected, processed, stored and archived.

If the school wishes to have the results of the GL NGRT standardised assessments, then they will need to sign a Data Sharing Agreement with BIT's Evaluation Team. FFT will share a contact name and email address for a named individual in your school with the EEF. For information on the purpose, processing, and retention of this data, please see their privacy notice.

The personal data collected for this project includes:

- Pupil data: name, date of birth, UPN, gender, national curriculum year, FSM, pupil premium, SEN, EAL, first language, ethnicity, class/teaching group, attainment data on reading skills and pupil participate in Reciprocal Reading
- Staff data: name, title, email and role

Parents/guardians of targeted pupils will be sent a letter from the school (provided by the BIT's Evaluation Team) outlining the data which will be processed and giving them the option of withdrawing their child's data from the study.

More information about the requirements for schools including information about data sharing is also summarised in the information sheet.

Who will the pupil - level data be shared with and why?

Personal data for pupils and staff will be shared by FFT with BIT's Evaluation Team to administer the GL NGRT reading tests and evaluate the impact of the intervention.

The personal data to be shared by FFT with BIT's Evaluation Team shall include:

- Pupil data: UPN, name, date of birth, gender, free school meal status, year group • Staff data: name, title, email and role

BIT's Evaluation Team will share pupils' UPNs and names with GL Assessment and Qa Research to administer the NGRT assessments. GL Assessment will provide the platform to administer the tests, while Qa Research will administer the tests to pupils in schools. BIT's Evaluation Team will use the pupil data to get further pupil information through the National Pupil Database, including pupils' KS2 National Test results and school census demographic data (see more information in the school information sheet which can be accessed here)
Personal data collected from the school will not be transferred outside of the UK by FFT or BIT's Evaluation Team (including BIT's delivery partners). In this research project, the FFT and BIT will all act as independent data controllers of the personal data that is provided by the school.

For the purposes of further research, pupil-level data will be shared to the EEF archive, held in the Office for National Statistics' Secure Research Service (ONS SRS). The data will be matched to the National Pupil Database (NPD) by the Department for Education (DfE). Pupils will subsequently be identified using the Pupil Matching Reference (PMR) an identifier meaningless to those without National Pupil Database (NPD) data. Pupil names and unique pupil identifiers (UPNs) are not held in the archive.

Sharing with other approved parties and further linking to NPD and other administrative data may take place during subsequent research. EEF will act as the data controller for the EEF data archive which is managed on their behalf by FFT who act as data processors.

The personal data used for research purposes will not be used for any other purpose. Outside of the archive, any personal data collected by FFT and BIT will be destroyed in accordance with GDPR regulations when it is no longer required, and no later than 31st July 2026 or, for the purpose of delivery of the Reciprocal Reading programme, when the school ceases to use the FFT Platform to support the schools' delivery of Reciprocal Reading.

Roles and Responsibilities

The Delivery Team (FFT) will:

- Recruit schools to the project.
- Provide a briefing on Reciprocal Reading for the headteacher and/or school lead coordinator for each school involved in the project.

- Provide two days face to face training for up to 3 staff from every school in the intervention group, together with the related resources required to deliver Reciprocal Reading.
- Provide two online support sessions (up to 90 minutes) to each of the intervention schools. • Provide schools with a screening tool (connected to the school MIS) to identify pupils to participate and report pupil attendance electronically.
- Collect school and pupil and school staff data (as described in this memorandum of understanding and the school information sheet).

The Evaluation team (BIT) will:

- Act as the first point of contact for any questions about the evaluation.
- Conduct the random allocation of schools to either the intervention group or control group.
- Provide information and withdrawal forms for schools to send to parents/ guardians. • Recruit for parent interviews, if required.
- Facilitate the pre- and post- reading tests supported by Qa Research in schools and ensure that all Qa Research staff working in schools have been DBS checked and completed safeguarding training. • Use pupil and school level data provided by FFT to obtain NPD data.
- Analyse the data from the project.

All schools will:

- Not participate in another EEF literacy randomised trial that would interfere with implementation of the intervention with Year 5 and 6 pupils during 2023/24 academic year.
- Before the end of academic year 2022 – 23, the school will assess Years 4/5 pupils in reading comprehension using the screening tool provided by FFT. This will help identify a group of 12-16 pupils (6 in Year 4 and 6 in Year 5) who will be suitable for a Reciprocal Reading programme in Years 5/6 in 2023 – 24.
- Provide the FFT and BIT's Evaluation team with pupil and school data requested and ensure all pupils participating in the trial complete all required GL NGRT standardised reading tests in summer term 2023 and summer term 2024.
- Deliver letters to parents/ guardians giving them information about the study and an opportunity to opt their child out of the data gathering process. They will inform the Evaluation Team of any responses arising and permit the publication of anonymised data collected.
- Agree to the Evaluation Team obtaining the evaluation cohorts' data from the National Pupil Database and will provide the UPNs to enable this to be achieved.

The Intervention Schools will:

- Receive a highly subsidised version of Reciprocal Reading and pay FFT £215+VAT to support the delivery of the programme
- Ensure the school lead coordinator attends an online briefing (60 minutes).
- Ensure that the 3 members of staff delivering Reciprocal Reading attend the two face-to-face training days.
- Provide the trained members of staff time to deliver the Reciprocal Reading programme in full to 12- 16 pupils in two groups in Year 5 and 6 (six pupils in each group) in the academic year 2023/24. • Take part in 2 online support sessions (90 minutes each) with the FFT Reciprocal Reading team (the 3 people delivering Reciprocal Reading to attend each session).
- Ensure that the online attendance registers are completed for all Reciprocal Reading sessions delivered in school using the system provided by FFT.

- Ensure teachers, at the earliest opportunity, notify FFT and the Evaluation Team if there are any issues which could prevent the effective implementation of Reciprocal Reading.
- Help the BIT evaluation team with the recruitment for parent interviews to speak to parents of pupils in the intervention about their home reading environments, if required.
- **Notify FFT and the Evaluation Team straight away if the school has to withdraw from the project for operational or other unavoidable reasons, and wherever possible still provide test data for the evaluation.**

The Control Schools will:

- Not participate in another EEF literacy randomised trial that would interfere with implementation of the intervention with Year 5 and 6 pupils during 2023/24 academic year.
- Before the end of academic year 2022 – 23, assess a group of pupils in Years 4/5 who are below the expected standard in reading comprehension using the online assessment provided by FFT. This will help identify a group of 12 pupils (6 in Year 4 and 6 in Year 5) who will be suitable for a Reciprocal Reading programme in Years 5/6 in 2023 – 24.
- Provide the FFT and Evaluation team with pupil and school data requested (see above) and will ensure all pupils participating in the trial complete all required standardised tests.
- Deliver letters to parents/guardians giving them information about the study and an opportunity to opt their child out of the data gathering process. They will inform the Evaluation Team of any responses arising and permit the publication of anonymised data collected.
- Agree to the Evaluation Team obtaining the evaluation cohorts' data from the National Pupil Database and will provide the UPNs to enable this to be achieved.
- Receive a payment of £1,000 from FFT in July 2024 for participating in the research project, once all the requirements have been completed.

Project Timeline

Date	Activities
	All schools
May - July 2023	Online briefings for school lead coordinators
June - July 2023	All year 4 & 5 pupils (academic year 2022/23) identified as being below the expected standard in reading comprehension will be assessed on their comprehension and decoding skills in May/June 2023. NGRT tests completed on 12 – 16 selected pupils

Date	Activities
September 2023	Schools which haven't completed NGRT tests do so Schools notified of allocation to intervention or control group.
Intervention schools only	
October - November 2023	Day 1 face-to-face training
October 2023	Schools begin delivering reciprocal reading sessions with Y5 and Y6 groups. Aim is for 2 sessions a week for at least 12 weeks. Complete attendance of sessions and submit to FFT.
December 2023 - January 2024	Online Meeting 1 with schools
January - February 2024	Day 2 face-to-face training
March - May 2024	Online Meeting 2 with schools
May - June 2024	All schools will have completed the reciprocal reading intervention programme.
All schools	
May - June 2024	Final assessments completed using NGRT
July 2024	Control schools can sign-up to online reciprocal reading training if they wish. Intervention schools can use reciprocal reading approaches and materials as they wish.
Spring 2025	Publication of evaluation report

School Agreement: to be signed by the Headteacher

I agree for my school to take part in the FFT Reciprocal Reading study and I accept the terms and conditions included within this Memorandum of Understanding.

If your school would like to participate in the FFT Reciprocal Reading study, please read, complete and electronically sign this form.

School name:	[Account.Name]
School DfE Number:	[Account.AccountNumber]
School URN:	[Account.URN__c]
School postcode:	[Account.BillingPostalCode]
Local authority:	[Account.Local_Authority_Parent__r.Name]
School phone number:	[Account.Phone]
Headteacher's full name:	[Decision Maker.FirstName][Decision Maker.LastName]
Headteacher's email:	[Decision Maker.Email]
School lead contact for FFT Reciprocal Reading	
School lead contact name:	
School lead contact email address:	

Face-to-face training: Requirement to travel to nearest available training location

The two face-to-face training session are mandatory. By signing this document, you are agreeing to travel to your nearest or most convenient venue location. All three colleagues rolling out the intervention in school must attend the training.	Bedford/Luton Black Country (West Midlands) Blackpool Bristol Cambridge/Peterborough Coventry Derby/Nottingham Durham/Sunderland Leeds Liverpool London Manchester Norwich/Ipswich South Yorkshire Stoke-on-Trent Swindon Teesside/Darlington

Invoice information	
Finance contact name:	
Finance email address:	
Invoice address (if different):	
Purchase order reference (if required):	
Fee for the school: *Only applicable if you are randomly selected as an 'intervention' school	£215 +VAT

Headteacher's signature and date:	

Order Confirmed by FFT Education *(FFT office use only)*

Authorised by	Alia Novak
Reference	[Account.AccountNumber]
Approved	Alia Novak

Thank you for agreeing to take part in this research. We look forward to working with you on this exciting project.

[SBQQ__PrimaryQuote__r.Name]

Appendix F: Randomisation code

```
# title: 1st batch randomisation
# author: "Neus Torres"
# QAer: Bram Reitsma
# date: "2023-08-13"
# project: EEF Reciprocal Reading

# This script randomises the first batch of schools (clusters) for EEF RR project and
# conducts some balance checks after merging the list of schools with the pupils.

# Before beginning: Please open the R project called "R project - randomisation"
# which is in the folder "Trial" of [Project data] UK001387.000 EEF Reciprocal Reading

rm(list = ls())

## 0. Load packages
library(dplyr)
library(tools)
library(readxl)

## 1 Load list of schools for 1st batch
 # These schools are the ones that have had the pupil assessments with the NGRT test
 # before 12 August 2023. There should be 225 schools

# Set the directory path
getwd()   # check directory is correct and leads to Trial folder
trialfolder <- getwd()
schools_list <- file.path(trialfolder, "Randomisation", "schools_for_batch1.csv")

# Load list
df <- read.csv(schools_list)
```

```
## 2. Check numbers are correct
num_obs <- nrow(df)
print(num_obs)
   # 225 rows

 # Rename variables to the names in the pupil dataset so they match
df <- df %>%
 rename(School.Name = Group, DFENumber=Custom1)

# Count the number of distinct values of the "Group" column
num_schools <- n_distinct(df$DFENumber)
print(num_schools)
   # 225 schools - correct
stopifnot(num_obs==num_schools)

## 3. Randomise the list of schools into treatment and control

# set random seed with today's date so randomisation is reproducible
set.seed(130823)

## Define the number of arms
n_arms <- 2

## Randomise schools in dataframe df using their DFE Number
df <- df %>%
 ## Create a variable called "random" with a random integer from 0 to 1
 mutate(random=runif(n=dplyr::n())) %>%
 ## Create a rank for that random integer
 mutate(rank=rank(random, ties.method = "random")) %>%
 ## Cut the rank sequence into 2 arms
 mutate(treatment_allocation=cut(rank, breaks=n_arms, labels=F))
```

```
## Check the numbers are alright
tab <- df %>%
 count(df$treatment_allocation) %>% print()
  ## 113 at treatment and 112 at control
df <- df %>%
 mutate(treatment_allocation = ifelse(treatment_allocation == 1, "Treatment", "Control"))

## 4. Merge with pupil data

## Load pupil data
pupildata <- file.path(trialfolder, "Working", "pupil_dataset.csv")
pupildata <- read.csv(pupildata)

## Sort by DFENumber
pupildata <- pupildata %>%
 arrange(DFENumber)

schools <- pupildata %>%
 count(School.Name)

## Check for duplicates before merging:
pupildata <- pupildata %>%
 group_by(Upn) %>%
 mutate(dupli_upn = n()-1) %>%
 ungroup()
print(table(pupildata$dupli_upn))

## Merge with the original data by the cluster variable
data <- df %>%
 dplyr::select(School.Name, DFENumber, num_pupils, batch, treatment_allocation)

mergeddata <- data %>% #BRAM: you don't need data %>% here (or you can remove x=data from the next line)
```

```
 left_join(x=data, y=pupildata, # master:data, using: pupildata
          by="DFENumber",     # merge by DFENumber
         multiple = "all")    # specify that each row of x will.match multiple rows on y

# BRAM: another way of doing this (inner join only keeps the combinations for which there's a match):
#mergeddata <- inner_join(pupildata,data,by="DFENumber")

## Calculate the frequency of missing values for each variable
missing <- mergeddata %>%
  summarise_all(~ sum(is.na(.)))
print(missing)

## Check for duplicates
mergeddata <- mergeddata %>%
  group_by(Upn) %>%
  mutate(dupli_upn = n()-1) %>%
  ungroup()
print(table(mergeddata$dupli_upn))
  # No duplicates

# Collapse the data frame to unique combinations of School.Name.x and School.Name.y
result <- mergeddata %>%
  distinct(School.Name.x, School.Name.y)
print(result)
    # they didn't match bc School.Name.x is in lowercase. Otherwise looks correct

# BRAM: to check this with all lower case and punctuation removed:
library(stringr)
temp <-mergeddata %>% select(contains("School.Name")) %>%
  mutate(across(everything(), ~ tolower(str_remove_all(., "[[:punct:]])))) %>% rowwise() %>%
  mutate(not_identical = ifelse(School.Name.x == School.Name.y, 0, 1))
sum(temp$not_identical) # BRAM - 0 so all rows are identical
```

```
## 5. Do some balance checks

head(mergeddata)


## Get pupil age from date of birth

# Convert Date.Of.Birth to Date object

mergeddata$dob <- as.Date(mergeddata$Date.Of.Birth)


# Calculate age

current_date <- as.Date("2023-08-15")

mergeddata$age <- as.integer(difftime(current_date, mergeddata$dob, units = "weeks") / 52.1775) # Average number of weeks in a year

# BRAM: One thing to be careful of is that this can chnage depending on when you run this.

# BRAM: I think it'd be better to change Sys.Date() to whatever the date is you do the final randomisation.

# I changed the Sys.Date to as.Date("2023-08-15")


## Tabulate all distinct observations from all variables

variables <- c("Eal", "FSM", "Gender", "Ethnicity", "SEND", "age", "YearGroup")

for (x in variables) {

  result_x <- mergeddata %>%

   group_by(.data[[x]]) %>%

   summarise(N = n())

  # Assign the result to a named object

  assign(paste("result_", x, sep = ""), result_x)

}


## Turn categorical variables into factors

variables <- c("Eal", "FSM", "Gender")

for (var in variables) {

 mergeddata[[var]] <- as.factor(mergeddata[[var]] )

}
```

```
for (var in variables) {
  print(paste("Levels for", var))
  print(levels(mergeddata[[var]]))
}

#mergeddata$eal <- ifelse(is.na(mergeddata$Eal), NA, ifelse(mergeddata$Eal == "EAL", 1, 0))
#mergeddata$fsm <- ifelse(is.na(mergeddata$FSM), NA, ifelse(mergeddata$FSM == "FSM", 1, 0))
#mergeddata$gender <- ifelse(is.na(mergeddata$Gender), NA, ifelse(mergeddata$Gender == "Female", 1, 0))
#mergeddata$send <- ifelse(is.na(mergeddata$SEND), NA, ifelse(mergeddata$SEND %in% c("EHCP", "School Support"), 1, 0))

# BRAM: a useful function for these kind of things is case_when. this will do the same as above:
mergeddata <- mergeddata %>%
  mutate(
    eal = case_when(
      is.na(Eal) ~ NA_integer_,
      Eal == "EAL" ~ 1,
      Eal == "Not EAL" ~ 0,
    ),
    fsm = case_when(
      is.na(FSM) ~ NA_integer_,
      FSM == "FSM" ~ 1,
      FSM == "Not FSM" ~ 0,
    ),
    gender = case_when(
      is.na(Gender) ~ NA_integer_,
      Gender == "Female" ~ 1,
      Gender == "Male" ~ 0,
    ),
    send = case_when(
      is.na(SEND) ~ NA_integer_,
      SEND %in% c("EHCP", "School Support") ~ 1,
      TRUE  ~ 0,
```

```
  ))


## Compare sample averages for treatment and control
variables <- c("eal", "fsm", "gender", "send", "YearGroup", "num_pupils", "age")
for (var in variables) {
 print(paste("Summary of", var))
 print(by(mergeddata[[var]], mergeddata$treatment_allocation, summary))
}
## Sample averages look close. Let's do a t-test for "Eal", "FSM", "Gender", "SEND" and baseline assessment

##Merge with the baseline results to see if sample is balanced across baseline attainment
## Load excel with results
baselinepath <- file.path(trialfolder, "Input", "Pupil data", "Baseline", "NGRT baseline results", 'NGRT_report_11082023.xlsx')
baseline <- read_excel(baselinepath, sheet = "StudentData", col_names = TRUE)

baseline <- baseline %>%
 rename(Upn = 'TW Unique ID', DFENumber = Custom1, baselineNGRT = 'NGRT - SAS')
baseline <- baseline %>%
 select(Upn, DFENumber, baselineNGRT)

mergeddata <- mergeddata %>%
 left_join(x=mergeddata, y=baseline, # master:mergeddata, using: baseline
        by="Upn",     # merge by DFENumber
        multiple = NULL,   # specify that each row of x will.match one row of y
        unmatched = "drop")

# BRAM: this will give the same result:
#temp <- mergeddata %>%
#  left_join(baseline, by="Upn")

# BRAM: you can add %>% print() to the pipe to immediately print, as such:
```

```
missing <- mergeddata %>%
  summarise(missing_count = sum(is.na(baselineNGRT))) %>% print()

# Pupils without a baseline score could be those pending a mop-up from the school

print(paste("Summary of baseline NGRT scores"))
print(by(mergeddata$baselineNGRT, mergeddata$treatment_allocation, summary))

# BRAM the tidverse way of doing this is:
treatment_group <- mergeddata %>% filter(treatment_allocation == "Treatment")
control_group <- mergeddata %>% filter(treatment_allocation == "Control")

variables_to_test <- c("eal", "fsm", "gender", "send", "baselineNGRT")
# BRAM: I would perhaps check difference in total number of pupils per group as well? # NEUS: added

results <- list()

for (var in variables_to_test) {
  t_test_result <- t.test(treatment_group[[var]], control_group[[var]])
  results[[var]] <- t_test_result
}

# Printing the t-test results & number of observations per group
for (var in variables_to_test) {
  cat("Variable:", var, "\n")
  cat("p-value:", results[[var]]$p.value, "\n")
  cat("Mean in Treatment:", mean(treatment_group[[var]], na.rm=T), "\n")
  cat("Mean in Control:", mean(control_group[[var]], na.rm=T), "\n\n")
  # Calculate and print total number of observations per group
  total_observations_treatment <- sum(!is.na(treatment_group[[var]]))
  total_observations_control <- sum(!is.na(control_group[[var]]))
```

```
  cat("Total Observations in Treatment:", total_observations_treatment, "\n")
  cat("Total Observations in Control:", total_observations_control, "\n")

  # Calculate and print difference in observations
  cat("Difference in Observations:", total_observations_treatment - total_observations_control, "\n\n")
}

  ## The only category where there is a statistical significant difference
  ## between treatment arms is SEND.

# Use only complete cases for baselineNGRT (or else t.test doesn't work)
# BRAM: the issue is not with t.test, which automaticallt removes NA's. the issue is with the ...
# BRAM: ... mean function. if you simply add "na.rm=T" to that function, you can just add "baselineNGRT" ...
# BRAM: ... to the list above, and not do any of the below :)
# NEUS: added!
  # Baseline attainment is balanced across treatment arms

## 6. Create a dataframe with the list of schools that will be sent to FFT
allocated_schools <- df %>%
  dplyr::select(School.Name, DFENumber, batch, treatment_allocation)

allocated_list <- file.path(trialfolder, "Randomisation", "treatment_allocation_schools_for_batch1.csv")
write.csv(allocated_schools, file = allocated_list, row.names = FALSE)
###

## 7. Add data on baseline completion to the list of schools that will be sent to FFT

## FFT needs to know assessment progress so they can delay communicating treatment assignment
## to schools that haven't completed their assessments

## Create a variable with % of completion rate of the baseline assessment
df <- df %>%
```

```
 mutate(completion_rate = round(num_tests/num_pupils * 100)) %>%
 dplyr::select(School.Name, DFENumber, SchoolAssessmentDate, num_pupils,
          num_tests, mopups_pending, completion_rate, batch, treatment_allocation)

tabulation <- df %>%
 group_by(mopups_pending, completion_rate) %>%
 summarise(count = n()) %>%
 print()

df <- df %>%
 mutate(delay_needed = case_when(
  mopups_pending > 3 ~ "Yes",
  TRUE ~ "No"
 ))

allocated_list <- file.path(trialfolder, "Randomisation", "treatment_allocation_schools_for_batch1_withinfo.csv")
write.csv(df, file = allocated_list, row.names = FALSE)

# title: 2nd batch randomisation
# author: "Neus Torres"
# QAer: Laure Bokobza
# date: "2023-09-27
# project: EEF Reciprocal Reading

# This script randomises the 2nd! batch of schools (clusters) for EEF RR project and
# conducts some balance checks after merging the list of schools with the pupils.

# Before beginning: Please open the R project called "R project - randomisation"
# which is in the folder "Trial" of [Project data] UK001387.000 EEF Reciprocal Reading

rm(list = ls())
```

```
## 0. Load packages
library(dplyr)
library(tools)
library(readxl)



## 1 Load list of schools for 1st batch
# These schools are the ones that have had the pupil assessments with the NGRT test
# after 12 August 2023. There should be 70 schools.
# In total, there will be 295 schools in the trial, including a school in Batch 1
# in the treatment arm that has withdrawn after randomisation.

# Set the directory path
getwd()   # check directory is correct and leads to Trial folder
# If not, you can use the following line:
# setwd("~/[insert your shortcut here]/BIT project data/[Project data] UK001387.000 EEF Reciprocal Reading/Trial")
trialfolder <- getwd()
schools_list <- file.path(trialfolder, "Randomisation", "Batch 2", "schools_for_batch2.csv")

# Load list
df <- read.csv(schools_list)

## 2. Check numbers are correct
num_obs <- nrow(df)
print(num_obs)
# 70 rows
# LBQA: very good habit of writing in comment the output you get so the QA-er can check they get the same! Me likee.

# Rename variables to the names in the pupil dataset so they match
df <- df %>%
  rename(School.Name = Group, DFENumber=Custom1)
```

```
# Count the number of distinct values of the "Group" column
num_schools <- n_distinct(df$DFENumber)
print(num_schools)
# 70 schools - correct
stopifnot(num_obs==num_schools)

## 3. Randomise the list of schools into treatment and control

# set random seed with today's date so randomisation is reproducible
set.seed(270923)

## Define the number of arms
n_arms <- 2

## Randomise schools in dataframe df using their DFE Number
df <- df %>%
  ## Create a variable called "random" with a random integer from 0 to 1
  mutate(random=runif(n=dplyr::n())) %>%
  ## Create a rank for that random integer
  mutate(rank=rank(random, ties.method = "random")) %>%
  ## Cut the rank sequence into 2 arms
  mutate(treatment_allocation=cut(rank, breaks=n_arms, labels=F))

  ## This creates "treatment_allocation" as a binary variable equal to 1 or 2
## Check the numbers are alright
tab <- df %>%
  count(df$treatment_allocation)
## 35 at treatment and 35 at control
## Batch 1 was 113 at treatment and 112 at control. 1 school in treatment dropped so
## now we have equal numbers. Hence I'm not going to constrain randomisation
## bc we will have even numbers in treatment & control overall
# LBQA: Ok -- all clear.
```

```
## Change the treatment numeric variable to a string Treatment if 1, Control if 2
df <- df %>%
  mutate(treatment_allocation = ifelse(treatment_allocation == 1, "Treatment", "Control"))

## 4. Merge with pupil data to do some balance checks

## Load pupil data
  ## Before loading the data, execute the R script in the Trial/Code/Pupil dataset for randomisation - Batch2.R
  ## to make sure that the pupil_dataset.csv is updated
pupildata <- file.path(trialfolder, "Working", "pupil_dataset_LBQA.csv")
pupildata <- read.csv(pupildata)

## Sort by DFENumber
pupildata <- pupildata %>%
  arrange(DFENumber)

schools <- pupildata %>%
  count(School.Name)

## Check for duplicates in Upn before merging:
pupildata <- pupildata %>%
  group_by(Upn) %>%
  mutate(dupli_upn = n()-1) %>%
  ungroup()
print(table(pupildata$dupli_upn)) # LBQA: all 0s -- ok.

## Merge with the original data by the cluster variable
data <- df %>%
  dplyr::select(School.Name, DFENumber, num_pupils, batch, treatment_allocation)
# LBQA: flag: this may be normal but 3 schools have NA for num_pupils
# the two that don't have any pupil data, and one other one (kings wood school and nursery) that seems to be only missing num_pupils?
```

```
# NT: I updated the school list - Now it should be only two
mergeddata <- data %>%
 left_join(x=data, y=pupildata, # master:data, using: pupildata
        by="DFENumber",     # merge by DFENumber
        multiple = "all")    # specify that each row of x will.match multiple rows on y

## Calculate the frequency of missing values for each variable
missing <- mergeddata %>%
 summarise_all(~ sum(is.na(.)))
print(missing)
 ## There are 6 schools that still haven't submitted their pupil data yet,
 ## so these figures make sense.
# LBQA: can you update with the number of schools that still haven't submitted their pupil data yet (2, I think?)
# NT: thanks, now updated - It reads 2 correctly
## Check for UPN duplicates if Upn is not missing
mergeddata <- mergeddata %>%
 group_by(Upn) %>%
 mutate(dupli_upn = ifelse(!is.na(Upn), n()-1, NA)) %>%
 ungroup()
print(table(mergeddata$dupli_upn))
# No duplicates

# Collapse the data frame to unique combinations of School.Name.x and School.Name.y
result <- mergeddata %>%
 distinct(School.Name.x, School.Name.y)
print(result)
# they didn't match bc School.Name.x is in lowercase. Otherwise looks correct

# LBQA check that this is it:
check <- mergeddata %>%
 mutate(School.Name.y = tolower(School.Name.y)) %>%
 mutate(nomatch_name = case_when(School.Name.x != School.Name.y ~ 0,
```

```
                School.Name.x == School.Name.y ~ 1,
                TRUE ~ NA_real_))
table(check$nomatch_name) # match for 988 schools; the 2 unmatched are because School.Name.y is missing


## 5. Do some balance checks
head(mergeddata)


## Get pupil age from date of birth
# Convert Date.Of.Birth to Date object
mergeddata$dob <- as.Date(mergeddata$Date.Of.Birth)
 # in the as.Date function the result is stored in days
 # (the number of days since the default origin in R, which is January 1, 1970.
# LBQA - style comment: as.Date() works just fine here, when working with dates generally in R I highly recommend the package lubridate


# Calculate age
current_date <- Sys.Date()
mergeddata$age <- as.integer((current_date - mergeddata$dob) / 365.25) # Divide by averge number of days in a year


#Note for myself: when you perform the subtraction operation between two Date objects in R,
# the result is returned as a difftime object by default. This object represents the time
# difference between two dates expressed in days. To get the age in years as a numeric value
# instead of a difftime object with "X days," you have to extract the numeric part of the difftime object.


## Tabulate all distinct observations from all variables
variables <- c("Eal", "FSM", "Gender", "Ethnicity", "SEND", "age", "YearGroup")
for (x in variables) {
 result_x <- mergeddata %>%
  group_by(.data[[x]]) %>%
  summarise(N = n())
 # Assign the result to a named object
 assign(paste("result_", x, sep = ""), result_x)
}
```

```
# LBQA - style comment: to tabulate (or cross-tabulate) variables I really like the tbl_summary function from gtsummary package

# install.packages("gtsummary")

library(gtsummary)

tbl_summary(mergeddata[, variables]) # gives you the same output but you don't have to create separate objects

# NT: love it, thanks

## Turn categorical variables into factors

variables <- c("Eal", "FSM", "Gender")

for (var in variables) {

  mergeddata[[var]] <- as.factor(mergeddata[[var]] )

}


for (var in variables) {

  print(paste("Levels for", var))

  print(levels(mergeddata[[var]]))

}


mergeddata$eal <- ifelse(is.na(mergeddata$Eal), NA, ifelse(mergeddata$Eal == "EAL", 1, 0))

mergeddata$fsm <- ifelse(is.na(mergeddata$FSM), NA, ifelse(mergeddata$FSM == "FSM", 1, 0))

mergeddata$gender <- ifelse(is.na(mergeddata$Gender), NA, ifelse(mergeddata$Gender == "Female", 1, 0))

mergeddata$send <- ifelse(is.na(mergeddata$SEND), NA, ifelse(mergeddata$SEND %in% c("EHCP", "School Support"), 1, 0))

# LBQA - style comment: nested ifelse commands are easier to read if you indent i.e. go to a new line when you start a new ifelse. See below

# mergeddata$eal <- ifelse(is.na(mergeddata$Eal), NA,

#                          ifelse(mergeddata$Eal == "EAL", 1, 0))


# LBQA - style comment: the tidyverse alternative to ifelse is case_when, which you can easily nest into mutate and have only one chunk of code. See below:

# mergeddata <- mergeddata %>%

#   mutate(eal = case_when(is.na(Eal) ~ NA_real_,

#                          Eal == "EAL" ~ 1,

#                          TRUE ~ 0),
```

```
#       fsm = case_when(is.na(FSM) ~ NA_real_,
#                   FSM == "FSM" ~ 1,
#                   TRUE ~ 0),
#       gender = case_when(is.na(Gender) ~ NA_real_,
#                     Gender == "Female" ~ 1,
#                     TRUE ~ 0),
#       send = case_when(is.na(SEND) ~ NA_real_,
#                   SEND %in% c("EHCP", "School Support") ~ 1,
#                   TRUE ~ 0))

## Compare sample averages for treatment and control
variables <- c("eal", "fsm", "gender", "send", "YearGroup", "num_pupils", "age")
for (var in variables) {
  print(paste("Summary of", var))
  print(by(mergeddata[[var]], mergeddata$treatment_allocation, summary))
}
## Sample averages look close. Let's do a t-test for "Eal", "FSM", "Gender", "SEND" and baseline assessment
# LBQA - style comment: you can also use tbl_summary or table1 here,
# both really easy to use and produce nice output, plus you can add t-test! See below:
tbl_summary(mergeddata[, c(variables, "treatment_allocation")],
        by = treatment_allocation) %>%
  add_p(test = everything() ~ "t.test")

##Merge with the baseline results to see if sample is balanced across baseline attainment
## Load excel with the latest results
baselinepath <- file.path(trialfolder, "Input", "Pupil data", "Baseline", "NGRT baseline results", 'NGRT_report_27092023.xlsx')
baseline <- read_excel(baselinepath, sheet = "StudentData", col_names = TRUE)

baseline <- baseline %>%
  rename(Upn = 'TW Unique ID', DFENumber = Custom1, baselineNGRT = 'NGRT - SAS')
baseline <- baseline %>%
  select(Upn, DFENumber, baselineNGRT)
```

```
# LBQA check missing values in baseline data
sum(is.na(baseline$Upn)) # 0
sum(is.na(baseline$DFENumber)) # 0
sum(is.na(baseline$baselineNGRT)) # 470 NA

# LBQA: check no UPN duplicates in baseline data
length(unique(baseline$Upn)) == nrow(baseline) # FALSE -- there are two duplicates
# identify duplicates
dupUPN_baseline <- baseline %>%
 group_by(Upn) %>%
 filter(n() > 1)
# two duplicate UPNs, one of them has two different values for baseline NGRT.
# Neus to check (they're not in the merged data so not sure it matters)

mergeddata <- mergeddata %>%
 left_join(x=mergeddata, y=baseline, # master:mergeddata, using: baseline
        by="Upn",     # merge by DFENumber
        multiple = NULL,    # specify that each row of x will.match one row of y
        unmatched = "drop")
missing <- mergeddata %>%
 summarise(missing_count = sum(is.na(baselineNGRT)))
print(missing)
# Pupils without a baseline score could be those pending a mop-up from the school
# or from schools pending assessment, which should be approx. = 29 schools * 14 = 406 pupils

print(paste("Summary of baseline NGRT scores"))
print(by(mergeddata$baselineNGRT, mergeddata$treatment_allocation, summary))
# There is a 2 point difference
# LBQA: I get mean treatment = 93.12 and mean control = 93.12, Neus to double-check this is what she gets
# NT: with the latest data from this morning I get 92.25 in control and 94.41 treatment
# Do t-tests
```

```
cleaned_data <- mergeddata %>%
 filter(!is.na(eal)) %>%
 filter(!is.na(fsm)) %>%
 filter(!is.na(gender)) %>%
 filter(!is.na(send))


 treatment_group <- cleaned_data[cleaned_data$treatment_allocation == "Treatment", ]
control_group <- cleaned_data[cleaned_data$treatment_allocation == "Control", ]

variables_to_test <- c("eal", "fsm", "gender", "send")

results <- list()

for (var in variables_to_test) {
 t_test_result <- t.test(treatment_group[[var]], control_group[[var]])
 results[[var]] <- t_test_result
}

# Printing the t-test results
for (var in variables_to_test) {
 cat("Variable:", var, "\n")
 cat("p-value:", results[[var]]$p.value, "\n")
 cat("Mean in Treatment:", mean(treatment_group[[var]]), "\n")
 cat("Mean in Control:", mean(control_group[[var]]), "\n\n")
}

#LBQA: I get NA for the mean in treatment and mean in control using this command, because there are NAs for your variables
# NT: I added lines 269 - 273
# Use only complete cases for baselineNGRT (or else t.test doesn't work)
cleaned_data <- mergeddata %>%
 filter(!is.na(baselineNGRT)) %>%
```

```
  mutate(baselineNGRT = as.numeric(baselineNGRT))

treatment_group <- cleaned_data[cleaned_data$treatment_allocation == "Treatment", ]
control_group <- cleaned_data[cleaned_data$treatment_allocation == "Control", ]

variables_to_test <- c("baselineNGRT")

results <- list()
for (var in variables_to_test) {
  t_test_result <- t.test(treatment_group[[var]], control_group[[var]])
  results[[var]] <- t_test_result
}
# Printing the t-test result for baseline assessment
for (var in variables_to_test) {
  cat("Variable:", var, "\n")
  cat("p-value:", results[[var]]$p.value, "\n")
  cat("Mean in Treatment:", mean(treatment_group[[var]]), "\n")
  cat("Mean in Control:", mean(control_group[[var]]), "\n\n")
}

# Baseline attainment is balanced across treatment arms - Difference is not v significative
# LBQA: OK.

## 6. Create a dataframe with the list of schools in Batch 2 that will be sent to FFT
allocated_schools <- df %>%
  dplyr::select(School.Name, DFENumber, batch, treatment_allocation, num_pupils, num_tests, mopups_pending)

allocated_list <- file.path(trialfolder, "Randomisation", "Batch 2", "treatment_allocation_schools_for_batch2_withinfo.csv")
write.csv(allocated_schools, file = allocated_list, row.names = FALSE)

## 7. Repeat the balance checks with the whole sample
  # IMport batch 1 allocation
```

```
batch1_list <- file.path(trialfolder, "Randomisation", "Batch 1", "treatment_allocation_schools_for_batch1.csv")
batch1 <-read.csv(batch1_list)

 # Merge batch 1 and 2
all_schools <- rbind(allocated_schools, batch1)

 # Merg full list with pupildata
mergeddata <- all_schools %>%
 left_join(x=all_schools, y=pupildata, # master:data, using: pupildata
        by="DFENumber",     # merge by DFENumber
        multiple = "all")    # specify that each row of x will.match multiple rows on y

head(mergeddata)

## Get pupil age from date of birth
mergeddata$dob <- as.Date(mergeddata$Date.Of.Birth)
current_date <- Sys.Date()
mergeddata$age <- as.integer((current_date - mergeddata$dob) / 365.25) # Divide by averge number of days in a year

## Tabulate all distinct observations from all variables
variables <- c("Eal", "FSM", "Gender", "Ethnicity", "SEND", "age", "YearGroup")
for (x in variables) {
  result_x <- mergeddata %>%
    group_by(.data[[x]]) %>%
    summarise(N = n())
  # Assign the result to a named object
  assign(paste("result_", x, sep = ""), result_x)
}

## Turn categorical variables into factors
variables <- c("Eal", "FSM", "Gender")
for (var in variables) {
```

```
  mergeddata[[var]] <- as.factor(mergeddata[[var]] )
}

for (var in variables) {
  print(paste("Levels for", var))
  print(levels(mergeddata[[var]]))
}

mergeddata$eal <- ifelse(is.na(mergeddata$Eal), NA, ifelse(mergeddata$Eal == "EAL", 1, 0))
mergeddata$fsm <- ifelse(is.na(mergeddata$FSM), NA, ifelse(mergeddata$FSM == "FSM", 1, 0))
mergeddata$gender <- ifelse(is.na(mergeddata$Gender), NA, ifelse(mergeddata$Gender == "Female", 1, 0))
mergeddata$send <- ifelse(is.na(mergeddata$SEND), NA, ifelse(mergeddata$SEND %in% c("EHCP", "School Support"), 1, 0))


## Compare sample averages for treatment and control
variables <- c("eal", "fsm", "gender", "send", "YearGroup", "age")
for (var in variables) {
  print(paste("Summary of", var))
  print(by(mergeddata[[var]], mergeddata$treatment_allocation, summary))
}
## Sample averages look very close. Let's do a t-test for "Eal", "FSM", "Gender", "SEND" and baseline assessment

##Merge with the baseline results to see if sample is balanced across baseline attainment
## Load excel with the latest results
baselinepath <- file.path(trialfolder, "Input", "Pupil data", "Baseline", "NGRT baseline results", 'NGRT_report_27092023.xlsx')
baseline <- read_excel(baselinepath, sheet = "StudentData", col_names = TRUE)

baseline <- baseline %>%
  rename(Upn = 'TW Unique ID', DFENumber = Custom1, baselineNGRT = 'NGRT - SAS')
baseline <- baseline %>%
  select(Upn, DFENumber, baselineNGRT)
```

```
mergeddata <- mergeddata %>%
 left_join(x=mergeddata, y=baseline, # master:mergeddata, using: baseline
        by="Upn",     # merge by DFENumber
        multiple = NULL,    # specify that each row of x will.match one row of y
        unmatched = "drop")
missing <- mergeddata %>%
 summarise(missing_count = sum(is.na(baselineNGRT)))
print(missing)
# Pupils without a baseline score could be those pending a mop-up from the school
# or from schools pending assessment, which should be approx. = 29 schools * 14 = 406 pupils

print(paste("Summary of baseline NGRT scores"))
print(by(mergeddata$baselineNGRT, mergeddata$treatment_allocation, summary))
# There os a 1 point difference
#LBQA: I get mean control = 93.98 and mean treatment = 93.62 here

# Do t-tests
treatment_group <- mergeddata[mergeddata$treatment_allocation == "Treatment", ]
control_group <- mergeddata[mergeddata$treatment_allocation == "Control", ]

variables_to_test <- c("eal", "fsm", "gender", "send")

results <- list()

for (var in variables_to_test) {
 # Remove missing values for the current variable
 treatment_var <- na.omit(treatment_group[[var]])
 control_var <- na.omit(control_group[[var]])

 # Perform t-test on non-missing values
 t_test_result <- t.test(treatment_var, control_var)
 results[[var]] <- t_test_result
```

```
}

# Printing the t-test results
for (var in variables_to_test) {
  cat("Variable:", var, "\n")
  cat("p-value:", results[[var]]$p.value, "\n")
  cat("Mean in Treatment:", mean(na.omit(treatment_group[[var]])), "\n")
  cat("Mean in Control:", mean(na.omit(control_group[[var]])), "\n\n")
}

## The only category where there is a statistical significant difference
## between treatment arms is SEND There are 20% of SEND students in treatment and 22% in Control.
## The importance seems marginal
# LBQA: I get p-value = 0.1667762 for SEND here, so not significant, and 20% in control and 25% in treatment.
# Neus to check
# LBQA: using tbl_summary instead
tbl_summary(mergeddata[, c(variables_to_test, "treatment_allocation")],
            by = treatment_allocation) %>%
  add_p(test = everything() ~ "t.test") # I get the same figures using this as using your function above.

# Use only complete cases for baselineNGRT (or else t.test doesn't work)
cleaned_data <- mergeddata %>%
  filter(!is.na(baselineNGRT)) %>%
  mutate(baselineNGRT = as.numeric(baselineNGRT))

treatment_group <- cleaned_data[cleaned_data$treatment_allocation == "Treatment", ]
control_group <- cleaned_data[cleaned_data$treatment_allocation == "Control", ]

variables_to_test <- c("baselineNGRT")

results <- list()
for (var in variables_to_test) {
```

```
  t_test_result <- t.test(treatment_group[[var]], control_group[[var]])
  results[[var]] <- t_test_result
}
# Printing the t-test result for baseline assessment
for (var in variables_to_test) {
  cat("Variable:", var, "\n")
  cat("p-value:", results[[var]]$p.value, "\n")
  cat("Mean in Treatment:", mean(treatment_group[[var]]), "\n")
  cat("Mean in Control:", mean(control_group[[var]]), "\n\n")
}

# Baseline attainment is balanced across treatment arms - Difference is not significant
# LBQA: Ok.

# Last step: add info on the mopups and send the allocation to FFT
allocated_info <- allocated_schools %>%
  left_join(x=mergeddata, y=baseline, # master:mergeddata, using: baseline
          by="Upn",     # merge by DFENumber
          multiple = NULL,    # specify that each row of x will.match one row of y
          unmatched = "drop")
missing <- mergeddata %>%
  summarise(missing_count = sum(is.na(baselineNGRT)))
print(missing)
```

Appendix G: Parent information sheet and withdrawal form – trial

Dear parent/guardian,

We are writing to let you know that your child has been selected to participate in a research project. The project is an evaluation of a reading programme called Reciprocal Reading. The reading programme is run by some specialists called the Fisher Family Trust, it is funded by the Education Endowment Foundation, and will be evaluated by the Behavioural Insights Team (BIT).

The aim of the evaluation is to understand how well this works to improve childrens' reading. To explore this, some schools across the country will run the programme and some schools will act as a comparison group. As part of this evaluation, your child:

- **Will complete two reading tests**; one in the autumn term and one in the spring. Note, the test is not part of the curriculum and will not affect your child's grades.
- **May receive two weekly additional reading sessions for 12 weeks**. If they do not receive these sessions, this is because your child's school is part of the comparison group needed to see how well the programme works.
- **May partake in a group interview** with other pupils to speak to us about the intervention. This will take place in school, during school hours with school staff supervision.

To evaluate the programme, your child's school will share data about your child with us (BIT), via the FFT. You can see all the information about how your child's data will be collected, used and protected in our privacy notice. There is also a section which explains what happens to your child's data if you withdraw them from the trial: https://www.bi.team/research-project-privacy-notice-reciprevaluationocal-reading-.

- If you would like any further information about the research, you can email rr-eval@bi.team

If you are happy for your child's data to be included in this evaluation, no action is required from you.

If you would prefer your child's data not to be shared, stored and used for this project, **complete the form below** and send it to us either:

- by sending a photo or PDF via email to rr-eval@bi.team with the subject "Data Withdrawal"; or
- by returning the slip to [SCHOOL TO INSERT TEACHER NAME] at the school.

You can ask to withdraw your child's data at any time until 31st of July 2024. At this point we will begin the process of analysing and deleting the personal data we hold.

You are under no obligation to agree to your child taking part in the research - sharing your child's data is entirely voluntary.

_ _

Reciprocal Reading Evaluation Withdrawal Form

Please complete this form if you **do not** wish for your child's data to be shared, stored, and used for Reciprocal Reading programme evaluation.

☐ I do NOT want data about my child to be collected, stored and used as part of the Reciprocal Reading Evaluation.

☐ I do NOT want my child to take part in a group interview.

Signature: _____

Name: _____

Your phone number: _____

Full name of your child: _____

Date: _____

Appendix H: Parent information sheet and withdrawal form – focus group

Dear parent/guardian,

As you know your child has been involved in a reading programme called Reciprocal Reading.

- Reciprocal Reading is being trialled in schools to find out whether it is effective in boosting children's reading skills. The programme is run by the Fisher Family Trust, it is funded by the Education Endowment Foundation, and is being evaluated by the Behavioural Insights Team (BIT), an independent research organisation.
- As part of the evaluation, BIT is conducting in-depth research in four case study schools, to better understand how the programme is being delivered.

Your child's school has been selected to be a case study.

- This means a researcher from BIT will be visiting the school to speak to teachers and pupils about their experience of the programme.
- This will involve your child taking part in an engaging and interactive group discussion with 6-7 other pupils, led by an experienced BIT researcher.
- It will take place in school, during school hours with school staff supervision.
- The discussion will be audio recorded and transcribed. The findings will feed into the final evaluation report. Please note everything your child says will be kept confidential and anonymous.

For more information about how your child's data will be collected, used and protected please see our privacy notice.

If you are happy for your child to take part in the focus group discussion, no action is required.

If you do not wish your child to take part, please complete the form below and return to [SCHOOL TO INSERT TEACHER NAME] at the school by [ADD DATE]

_ _

Reciprocal Reading Evaluation - Focus Group Withdrawal Form

☐ I do NOT want my child to take part in a group interview.

Signature: _____

Name: _____

Full name of your child: _____

Date: _____

Appendix I: BIT privacy notice

Introduction

The Behavioural Insights Team (BIT) is an independent research company. We are conducting an evaluation of the Reciprocal Reading programme, which is being run with pupils in schools across England. This work is important to help us understand the impact of this programme and any improvements that can be put into place to make it better, for other schools and pupils in future. This research is being funded by the Education Endowment Foundation (EEF), and conducted in collaboration with the Fisher Family Trust (FFT) who provide the Reciprocal Reading programme to schools.

This privacy notice sets out how BIT collects and uses the personal data of school staff, pupils and parents ('participants') for the purposes of this evaluation.

Contact details

Behavioural Insights Ltd (the legal name of Behavioural Insights Team (BIT)) is the controller and is responsible for your personal data collected in connection with this evaluation. This notice applies to the personal data we collect directly from participants and personal data which is provided to us by third parties. Where we collect personal data from participants directly, please make sure that any personal details you provide are accurate and up to date, and let us know about any changes as soon as possible.

BIT will also receive participant data from FFT, the organisation delivering the Reciprocal Reading programme, who will act as a separate data controller. Please refer to their privacy notice for more information about their privacy practices.

We have appointed a Data Protection Officer (DPO) who is responsible for overseeing questions in relation to this privacy notice. If you have any questions about this privacy notice, including any requests to exercise your legal rights in relation to your personal data, please contact the DPO:

Post: Behavioural Insights Ltd, 58 Victoria Embankment, London, EC4Y 0DS.

Email: dpo@bi.team. You also have the right to make a complaint at any time to the Information Commissioner's Office (ICO), the UK supervisory authority for data protection issues (www.ico.org.uk). We would, however, appreciate the chance to deal with your concerns before you approach the ICO so please contact us in the first instance.

What personal data will we collect?

Pupil data

We will collect personal data from pupils involved in the research including:

- Unique Pupil Number (UPN, identifies the pupil in the National Pupil Database)
- First name
- Last name
- Gender
- Date of birth
- School name
- Year group

- New Group Reading Test (NGRT) pre-intervention and post-intervention scores
- Key Stage 2 SATs Reading scores
- Free school meal (FSM) status
- Ethnicity*
- Whether the pupil speaks English as an Additional Language
- Special Educational Needs status and category*
- Type of special educational need*
- Focus group discussion data - qualitative data gathered in focus groups on attitudes to the reciprocal reading intervention
- Reciprocal reading session attendance data - data on the number and length of reciprocal reading interventions attended
- Teachers' perceptions of their home reading environment (available resources, parental attitudes to reading, exposure to reading) - qualitative information will be gathered from teachers and used to contextualise responses from pupils (this only applies to pupils who are taking part in a focus group).

*Some of this data – ethnicity and data about health conditions – constitutes 'special category data' under data protection laws, and additional protections will apply to our collection and use of this data. This information is vital for our analysis, to assess the reach and possible differential effects of the programme. Reporting on these fields for the purpose of our research will be in an aggregated format only.

Pupils' parent data

We will collect personal data from parents of the pupils involved in the research including:

- First name
- Last name
- Email address
- Telephone number
- Data on socio-economic background (FSM status) of their children
- Interview data - qualitative data on attitudes to reading and self-reported reading behaviours in the home

School staff data

We will collect personal data from school staff involved in the research including:

- First name
- Last name
- Work email address
- Work telephone number
- Survey data - quantitative and qualitative data on attitudes to the reciprocal reading intervention
- Interview data - qualitative data on attitudes to the reciprocal reading intervention
- Reasons for a school's withdrawal from the trial and other specific circumstances within the school or concerning school staff that affect the school's ability to participate in the Reciprocal Reading programme or in the evaluation activities. This includes information collected from emails sent by school staff to FFT on these topics.

What do we do with information we collect?

BIT is collecting your personal data to deliver and manage an independent evaluation of the Reciprocal Reading programme.

As part of this evaluation, BIT will run a trial in approximately 300 primary schools in England. This trial will involve randomly selecting half of these schools to receive the programme ('intervention schools') and half to not receive it ('control schools').

Selected pupils in the intervention schools will receive twice weekly 30 minute reading sessions, led by a Teacher/Teaching Assistant (TA) - for a minimum of 12 weeks during the Spring Term of 2024. The control schools will carry on with their business as usual.

At the end of the trial, BIT will compare the reading outcomes of pupils in the intervention schools to those in the control schools to work out the impact of the Reciprocal Reading programme. BIT will also assess how the impact (if any) is achieved.

What is our lawful basis for processing your personal data?

Data protection laws require us to meet certain conditions before we are allowed to use your data in the manner described in this notice, including having a lawful basis for the processing.

For all information collected, BIT is relying on the lawful basis of:

LEGITIMATE INTERESTS: Our lawful basis for processing the personal data listed above is legitimate interests (as per Article 6 (1) (f) of the GDPR) and we have considered that participants' interests and fundamental rights do not override those legitimate interests). It is necessary in BIT's 'legitimate interests' to process the personal data identified above in order to conduct an evaluation of the Reciprocal Reading programme that has been commissioned by the EEF. The research project fulfils BIT's core business aims including undertaking research, evaluation and information activities in sectors that will deliver social impact.

For special category data (see above), we also rely on the following lawful basis:

SCIENTIFIC RESEARCH PURPOSES: Our processing is necessary for scientific research purposes in the public interest, subject to suitable and specific measures to safeguard the fundamental rights and the interests of the data subject, as required by applicable laws.

Who has access to your information?

Your information will be accessed by a limited number of researchers and advisors in BIT's project team working on this project.

BIT may disclose participants' information to third parties in connection with the purposes of processing your personal data set out in this notice. These third parties may include:

- regulators, law enforcement bodies and the courts, in order to comply with applicable laws and regulations, assist with regulatory enquiries, and cooperate with court mandated processes, including the conduct of litigation;

- suppliers, research assistants and sub-contractors who may process information on behalf of BIT e.g. cloud services to store data, transcription services, and survey platforms. These third parties are known as data processors and when we use them we have contractual terms and policies and procedures in place to ensure that your personal data is protected. This does not always mean that they will have access to information that will directly identify participants as we will share anonymised or pseudonymised data only wherever possible. We remain responsible for participants' personal information as the controller; and
- any third party to whom we are proposing to sell or transfer some or all of our business or assets.

BIT will share participants' data with the following organisations:

- Qa Research - An independent provider contracted to administer the New Group Reading Test (NGRT) in partner schools.
- GL Assessments - An independent provider which will provide the platform to run the NGRT tests.
- Schools participating in the research, which will receive pupil NGRT scores at the end of the study.
- FFT, which will receive pupil NGRT scores at the end of the study.
- The Department for Education (DfE) - National Pupil Database (NPD) team.
- The Office for National Statistics - Secure Research Service (ONS SRS) - We will use the ONS SRS to analyse the trial data securely when it is matched to NPD data.
- At the end of the evaluation, the data will be shared with the EEF and FFT Education (EEF's data processor for their archive). All EEF trial data is stored in the EEF data archive, held within the ONS SRS. The archive does not contain direct identifiers like pupil name, contact details and date of birth, but does hold a Pupil Matching Reference (PMR). The PMR is used for further matching to the NPD and other administrative datasets that may be required as part of subsequent research. We will not use pupil names or school names in any report arising from the research. For information on how the EEF will use and protect participants' data, please see their Data protection statement regarding EEF evaluations.

We may also disclose participants' personal information if required by law, or to protect or defend ourselves or others against illegal or harmful activities, or as part of a reorganisation or restructuring of our organisations.

International transfers

We may share your personal information with SmartSurvey, our technical service provider, for the purpose of survey administration.

In such circumstances, your information may be transferred outside of the UK or EEA. In this case, we will ensure appropriate safeguards are implemented and participants' are fully informed of the transfer (unless it is otherwise prohibited by law to inform you). If you require further information about this, you can request it from the Data Protection Officer.

Security

We take reasonable steps to protect participants' personal information and follow procedures designed to minimise unauthorised access, alteration, loss or disclosure of participants' information.

Taking into account the state of the art, the costs of implementation and the nature, scope, context and purposes of processing as well as the risk of varying likelihood and severity for the rights and freedoms of natural persons, we implement appropriate technical and organisational measures to ensure a level of security appropriate to the risk of processing.

We ensure that those who have permanent or regular access to personal data, or that are involved in the processing of personal data, are trained and informed of their rights and responsibilities when processing personal data. We provide such access on a need-to-know basis, and have measures in place which are designed to remove that access once it is no longer required.

Physical personal devices used by BIT are encrypted to protect your data, and confidential hard copy data (including special category data) is kept in locked rooms or cabinets.

We have put in place procedures to deal with any suspected personal data breach and will notify participants' and any applicable regulator of a breach where we are legally required to do so.

Data retention

We will only retain participants' personal data for as long as necessary to fulfil the purposes we collected it for, including for the purposes of satisfying any legal, accounting, or reporting requirements. When it is no longer necessary to retain participants' personal data, it will be securely deleted.

To determine the appropriate retention period for personal data, we consider the amount, nature, and sensitivity of the personal data, the potential risk of harm from unauthorised use or disclosure of personal data, the purposes for which we process personal data and whether we can achieve those purposes through other means, and the applicable legal requirements.

Taking the above factors into consideration, and save as set out below in relation to teacher data, our anticipated date of deletion for participants' personal data is September 2025 (3 months after project completion, defined as the date that the final report is signed off by the EEF).

We may retain the names and work contact details of school staff beyond this date for the purposes of inviting schools or their staff to participate in future similar projects at BIT. If you do not wish to be contacted for this reason, please email dpo@bi.team.

We may keep research consent forms which contain personal information for a number of years after the research has been completed in order to meet legal and statutory requirements and/or because this is a requirement of the research's funder.

In some circumstances, we will retain an anonymised dataset (so that it can no longer be associated with participants) for research or statistical purposes, in which case we may use this information indefinitely without further notice to participants.

Withdrawing participation

At any point until 31st July 2024, you can withdraw your child's participation from the trial. If you withdraw your child from the trial before any of their personal data is shared with BIT - then FFT Literacy and schools will not share your child's personal data. The pupil will still partake in Reciprocal Reading sessions, but their data will not be collected, and they will not be tested using the NGRT reading tests.
Should we be informed that you would like to withdraw your child from the trial after some personal data has been collected we will not collect any additional personal data for your child and will, where possible, delete all of your child's personal data collected up to that point (other than interview responses that have already been analysed and anonymised, which will be handled in accordance with the remainder of this privacy notice).

Your legal rights

Under certain circumstances, you have rights under data protection laws in relation to your personal data, including rights to:

- Request access to your personal data: This enables you to receive a copy of the personal data we hold about you and to check we are lawfully processing it.
- Request correction of your personal data: This enables you to have any incomplete or inaccurate data we hold about you corrected.
- Request erasure of your personal data: This enables you to ask us to delete or remove personal data where there is no good reason for us continuing to process it.
- Object to processing of your personal data: For example, you can object where we are relying on a legitimate interest (or those of a third party) and there is something about your particular situation which makes you want to object to processing on this ground as you feel it impacts on your fundamental rights and freedoms.
- Request restriction of processing your personal data: This enables you to ask us to suspend the processing of your personal data.
- Data portability: Where the processing takes place on the basis of your consent or contract, and is carried out by automated means, you have the right to request that we provide your personal data to you in a machine-readable format, or transmit it to a third party data controller, where technically feasible.
- Withdraw consent to the processing of your personal data: This applies where we have relied on consent to process personal data. Please note that withdrawal of consent will not affect the lawfulness of any processing carried out before withdrawing your consent.
- Not be subject to decisions based purely on automated processing where it produces a legal or similarly significant effect on you. Please note that BIT does not engage in automated decision making without manual intervention in its research projects.

If you wish to exercise any of the rights set out above, please contact the Data Protection Officer with your specific request by email to: dpo@bi.team.

It is important to understand that the extent to which these rights apply to research will vary and that in some circumstances your rights may be restricted.

Ordinarily, you will not have to pay a fee to access your personal data (or to exercise any of the other rights). However, we may charge a reasonable fee if your request is clearly unfounded, repetitive or excessive. Alternatively, we may refuse to comply with your request in these circumstances.

We may need to request specific information from you to help us confirm your identity and ensure your right to access your personal data (or to exercise any of your other rights). This is a security measure to ensure that personal data is not disclosed to any person who has no right to receive it. We may also contact you to ask you for further information in relation to your request to speed up our response.

We try to respond to all legitimate requests within one month. Occasionally it may take us longer than a month if your request is particularly complex or you have made a number of requests. In this case, we will notify you and keep you updated.

Please also note that we can only comply with a request to exercise your rights during the period for which we hold personal information that directly identifies you. If we have only collected pseudonymised information (e.g. where we have not collected any names or contact details) or personal data has been irreversibly anonymised and has become part of the research data set, it will not be possible for us to comply.

Changes to this Notice

We may change this Privacy Notice from time to time. If we make any significant changes in the way we treat your personal information we will make this clear by updating the notice on the project website https://www.bi.team/research-project-privacy-notice-reciprocal-reading-evaluation or by contacting you directly.

Appendix J: Pre-post training analysis – knowledge quiz results

Table 1: Pre-post training analysis – knowledge quiz results

Knowledge question	% correct at pre-training	% correct at post-training	% increase
Which of the following can be used to challenge children in Reciprocal Reading?	26%	68%** [60%-75%]	42%
Which of the following could NOT be used for determining text difficulty for Reciprocal Reading?	15%	48%** [40%-57%]	33%
How could you increase the level of challenge during the Reciprocal Reading session?	60%	80%** [75%-85%]	20%
Which of the following techniques could be used as effective follow-up activities?	9%	58%** [47%-68%]	49%
Why is it useful to 'review and reflect' at the end of the session?	53%	59% [53%-66%]	6%
Average	33%	63%	30%

** = p < .001. Range in square brackets.

Appendix K: Deviations from the protocol

Protocol section	Specified analysis	Changes made	Justifications for changes
Outcome measures	KS2 Reading score to be measured using the KS2_KS2READSCORE variable from the NPD.	KS2 Reading score was measured using the KS2_READSCORE variable from the NPD.	The KS2_KS2READSCORE was not available and the differences between the two are minimal.
Statistical analysis	Not specified	*New sensitivity analysis: IT issues* We re-estimate the primary outcome model excluding all pupils from both treatment arms that were flagged by Qa Research as having experienced significant IT issues during endline testing. We compare the results with the main model to assess the extent to which these issues could have affected the main results. *New sensitivity analysis: excluding month of test* We re-estimate the primary outcome model excluding the month of endline testing as a covariate and compare the results with the main model to assess the risk of bias from the differences in the month of test.	IT issues were unexpected and we wanted to check whether they influenced the results. We excluded month of test as a covariate to assess the risk of bias from the differences in the month of test.

Appendix L: Primary and secondary analysis code

```
## ----setup2----------------------------------------------------------------------------------------------------------

## Load packages
library(tidyverse)
library(stringdist)
library(glm2) # for all regression models if needed
library(broom)
library(sandwich) # for clustered standard errors
library(zoo)
library(lmtest)
library(rmarkdown)

## Clean current working environment
rm(list = ls())

## Check current directory - It should be the same folder of the quarto document
getwd()



## ----load_data-------------------------------------------------------------------------------------------------------
## Load main dataset
#DELETED FOR CLEARANCE

master <- master %>% select(-...1)



## ----analysis_function-----------------------------------------------------------------------------------------------
```

```
 ### Compute total observations per treatment arm (will be needed later)
N_treatment <- sum(master$treatment_allocation=="Treatment", na.rm = TRUE)
N_control <- sum(master$treatment_allocation=="Control", na.rm = TRUE)

 ### Transform treatment variable into factor ###
master <- master %>%
 mutate(
  treatment_binary = if_else(treatment_allocation=="Treatment", 1, 0)
 ) %>%
 relocate(treatment_binary, .after = treatment_allocation)

master %>% count(treatment_allocation, treatment_binary)

 ## Set up the function
treatment_effect_ols <- function(data, y, treatment_variable, treatment_binary, covariates = NULL, cluster) {

 ## Set regression formula
 regression_formula <- as.formula(
  paste(y, "~", treatment_binary,
      if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else "")
 )

 print(regression_formula)

 ## Run OLS regression - use glm2 package
 model <- glm(regression_formula,
         data = data,
         family = gaussian # Use linear regression - OLS
         )
```

```
## Compute Clustered SE by school ID using sandwich package
clustered_se <- vcovCL(model, cluster = data[[cluster]], type = "HC1")

## Extract treatment coefficients and p-value from matrix
coef_test <- coeftest(model, vcov = clustered_se)
print(rownames(coef_test))

  ## Inspect matrix
print("Regression results:")
print(coef_test)

treatment_coef <- coef_test[2, 1]

p_value <- coef_test[2, 4]

se_clustered <- coef_test[2, 2]

## Compute the 95% Confidence internal for the treatment coefficient
conf_int95 <- treatment_coef + c(-1.96, 1.96)*se_clustered

## Compute the raw means, CI of the means and unadjusted difference
stats_df <- data %>%
  group_by(!!sym(treatment_variable)) %>%
   # Ignore missing values (we'll only use observations with non-missing values, as otherwise they are dropped from the regression)
  summarise(
  mean_y = mean(!!sym(y), na.rm = TRUE),
  sd_y = sd(!!sym(y), na.rm = TRUE),
  n = n(),
```

```
  se = sd_y / sqrt(n),
  ci_lower = mean_y - 1.96 * se,
  ci_upper = mean_y + 1.96 * se,
  .groups = "drop")

  # Extract raw means and  CI
mean_control <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Control") %>%
  pull(mean_y)

mean_treatment <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Treatment") %>%
  pull(mean_y)

  unadjusted_diff <- mean_treatment - mean_control

ci_lower_control <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Control") %>%
  pull(ci_lower)

ci_lower_treatment <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Treatment") %>%
  pull(ci_lower)

ci_upper_control <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Control") %>%
  pull(ci_upper)

ci_upper_treatment <- stats_df %>%
```

```
 filter(stats_df[[treatment_variable]]=="Treatment") %>%
 pull(ci_upper)

 ## Extract sample sizes
n_control <- sum(data[[treatment_variable]]=="Control", na.rm = TRUE)
n_treatment <- sum(data[[treatment_variable]]=="Treatment", na.rm = TRUE)
total_n <- n_control + n_treatment

## Extract missing sizes
n_control_missing <- N_control - n_control
n_treatment_missing <- N_treatment - n_treatment

## Compute variance of Y by treatment group
var_df <- data %>%
 group_by(!!sym(treatment_variable)) %>%
 summarise(variance = var(!!sym(y), na.rm = TRUE), .groups = "drop")

head(var_df)

## Extract variances
var_control <- var_df %>%
 filter(var_df[[treatment_variable]]=="Control") %>%
 pull(variance)

var_treatment <- var_df %>%
 filter(var_df[[treatment_variable]]=="Treatment") %>%
 pull(variance)

## Compute pooled SD
```

```
pooled_sd <- sqrt(((n_control - 1)* var_control + (n_treatment-1)* var_treatment) / (n_control + n_treatment - 2))

pooled_var <- pooled_sd^2

## Compute hedges G
hedges_g <- treatment_coef / pooled_sd

## Compute 95% CI for hedges G
hedges_g_ci <- conf_int95 / pooled_sd

## Print results
cat("\n### Results for outcome: ", y, "\n")
cat(" - Unadjusted difference in means: ", unadjusted_diff, "\n")
cat(" - N treatment: ", n_treatment, "\n")
cat(" - N treatment (missing): ", n_treatment_missing, "\n")
cat(" - N control: ", n_control, "\n")
cat(" - N control (missing): ", n_control_missing, "\n")
cat(" - Total N: ", total_n, "\n")
cat(" - Mean treatment: ", mean_treatment, "\n")
cat(" - Mean control: ", mean_control, "\n")
cat(" - 95% CI lower bound treatment: ", ci_lower_treatment, "\n")
cat(" - 95% CI upper bound treatment: ", ci_upper_treatment, "\n")
cat(" - 95% CI lower bound control: ", ci_lower_control, "\n")
cat(" - 95% CI upper bound control: ", ci_upper_control, "\n")

cat(" - Treatment coefficient: ", treatment_coef, "\n")
cat(" - p-value of treatment coefficient: ", p_value, "\n")
cat(" - 95% CI for treatment coefficient: ", conf_int95, "\n")
cat(" - Variance of ", y, "by treatment group:\n")
```

```
  cat("    - Treatment: ", var_treatment, "\n")
  cat("    - Control: ", var_control, "\n")
  cat(" - Pooled SD: ", pooled_sd, "\n")
  cat(" - Pooled variance: ", pooled_var, "\n")
  cat(" - Hedges G: ", hedges_g, "\n")
  cat(" - 95% CI for the Hedges G: [", hedges_g_ci[1], ",", hedges_g_ci[2],"]", "\n")

  ## Return results as a list
  return(list(
    unadjusted_diff = unadjusted_diff,
    treatment_coef = treatment_coef,
    p_value = p_value,
    conf_int95 = conf_int95,
    pooled_sd = pooled_sd,
    hedges_g = hedges_g,
    hedges_g_ci = hedges_g_ci,
    variance = var_df
  ))

}



## ----primary_analysis--------------------------------------------------------------------------------------------------

  ### Get only observations in the analysis
primary_df <- master %>%
  filter(In_Primary_analysis == 1)

    ## Call the function - SUbstitute for the right covariates in the model
```

```
results_primary_analysis <- treatment_effect_ols(
  data = primary_df, # dataframe to be used
  y = "Overall_reading_score_el", # outcome
  treatment_variable = "treatment_allocation",
  treatment_binary = "treatment_binary",  # treatment in binary format for regression
    # List of covariates:
  covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin", "YearGroup", "month_of_test"),
  cluster = "School_ID" # cluster variable
)

primary_df_reduced <- master %>%
  filter(!is.na(Overall_reading_score) & !is.na(Overall_reading_score_el) & !is.na(treatment_binary) &
!is.na(randomisation_batch))

results_primary_analysis_reduced <- treatment_effect_ols(
  data = primary_df_reduced, # dataframe to be used
  y = "Overall_reading_score_el", # outcome
  treatment_variable = "treatment_allocation",
  treatment_binary = "treatment_binary",  # treatment in binary format for regression
    # List of covariates:
  covariates = c("randomisation_batch", "Overall_reading_score"),
  cluster = "School_ID" # cluster variable
)

rm(primary_df_reduced)

  ### Sensitivity check: drop month of test as covariate
results_primary_analysis_nomonths <- treatment_effect_ols(
  data = primary_df, # dataframe to be used
  y = "Overall_reading_score_el", # outcome
```

```
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",
  # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin", "YearGroup"),
 cluster = "School_ID" # cluster variable
)



## ----secondary_pc----------------------------------------------------------------------------------------------------
 ## Repeat analysis for secondary outcome: Passage comprehension

 ### Get only observations in the analysis
master <- master %>%
 mutate(
  In_PC_analysis = if_else(
   !is.na(treatment_allocation) & !is.na(randomisation_batch) &
    !is.na(PC_score) & !is.na(PC_score_el) & !is.na(YearGroup) & !is.na(gender_admin) &
    !is.na(eal_admin) & !is.na(month_of_test), 1, 0)
 )

# LB QA flag: note fsm_admin is included in the sample definition here (and presumably in the covariates below) NTB: removed

secondary_PC_df <- master %>% filter(In_PC_analysis == 1)

  ## Call the function - SUbstitute for the right covariates in the model
results_secondary_analysis_pc <- treatment_effect_ols(
 data = secondary_PC_df,
 y = "PC_score_el", # outcome
 treatment_variable = "treatment_allocation",
```

```
  treatment_binary = "treatment_binary",
    # List of covariates:
  covariates = c("randomisation_batch", "PC_score", "eal_admin", "gender_admin", "YearGroup", "month_of_test"),
  cluster = "School_ID" # cluster variable
)




## ----self-selection_PC-----------------------------------------------------------------------------------------------
  ### % of pupils with NGRT overall score that are missing the PC score (they couldn't reach required level)
master <- master %>%
  mutate(
    missing_PC_el = is.na(PC_score_el), ## missing endline PC
    missing_PC = is.na(PC_score))      ## missing baseline PC


  ## endline
print("% of pupils with NGRT overall score that are missing the PC score, endline")
master %>%
  filter(!is.na(Overall_reading_score_el)) %>%
  count(missing_PC_el) %>%
  mutate(
    perc = round(n/sum(n)*100,1)
  )


  ## baseline
print("% of pupils with NGRT overall score that are missing the PC score, baseline")
master %>%
  filter(!is.na(Overall_reading_score)) %>%
  count(missing_PC) %>%
```

```
 mutate(
  perc = round(n/sum(n)*100,1)
 )

### % of pupils in the primary analysis that are missing the PC score (they couldn't reach required level)
print("% of pupils in primary analysis that are missing the PC score")
master %>%
 filter(In_Primary_analysis==1) %>%
 count(missing_PC) %>%
 mutate(
  perc = round(n/sum(n)*100,1)
 )

 ### Number of pupils in treatment and control group that have no PC at baseline, but do have one at endline
master <- master %>%
 mutate(missing_PC_prepost = is.na(PC_score) & !is.na(PC_score_el))

print("Number of pupils in treatment and control group that have no PC at baseline, but do have one at endline")
master %>%
 group_by(treatment_allocation) %>%
 count(missing_PC_prepost) %>%
 mutate(
  perc = round(n/sum(n)*100,1)
 )

 ### Logistic regression to see if likelihood of missing PC depends on treatment assignment, controlling for baseline sentence completion

 ## Run logistic regression - use glm2 package
 model_selfselection <- glm(missing_PC_el ~ treatment_binary + SC_score,
```

```
          data = master,
          family = binomial(link="logit"), # Use logit
          subset = !is.na(Overall_reading_score)
          )

summary(model_selfselection)



## ----secondary_sc----------------------------------------------------------------------------------------------------
 ## Repeat analysis for secondary outcome: sentence completion

 ### Get only observations in the analysis
   ### this is necessary because some statistics have to be computed from the analysis sample
master <- master %>%
 mutate(
   In_SC_analysis = if_else(
    !is.na(treatment_allocation) & !is.na(randomisation_batch) &
     !is.na(SC_score) & !is.na(SC_score_el) & !is.na(YearGroup) & !is.na(gender_admin) &
     !is.na(eal_admin) & !is.na(month_of_test), 1, 0)
 )

secondary_SC_df <- master %>%
 filter(In_SC_analysis == 1)

   ## Call the function - SUbstitute for the right covariates in the model
results_secondary_analysis_SC <- treatment_effect_ols(
 data = secondary_SC_df, # dataframe to be used
 y = "SC_score_el", # outcome
 treatment_variable = "treatment_allocation",
```

```
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = c("randomisation_batch", "SC_score", "eal_admin", "gender_admin", "YearGroup", "month_of_test"),
 cluster = "School_ID" # cluster variable
)

rm(secondary_SC_df)



## ----ks2_reading----------------------------------------------------------------------------------------------------------
# LB QA note: Equation 5, p.12 in SAP

 ### Get only observations in the analysis
master <- master %>%
 mutate(
  In_ks2_analysis = if_else(
   !is.na(treatment_allocation) & !is.na(randomisation_batch) &
    !is.na(KS2_READSCORE_num) & !is.na(Overall_reading_score), 1, 0)
 )
master %>% count(KS2_READSCORE, KS2_READSCORE_num)

ks2_df <- master %>%
 filter(In_ks2_analysis == 1)

 ### Compute results of main model - See page 9 of the SAP

  ## Call the function - SUbstitute for the right covariates in the model
results_ks2_analysis <- treatment_effect_ols(
 data = ks2_df, # dataframe to be used
```

```
 y = "KS2_READSCORE_num", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score"), # covairates in Equation 5
 cluster = "School_ID" # cluster variable
)

summary(ks2_df$KS2_READSCORE_num)
table(ks2_df$KS2_READSCORE_num)
rm(ks2_df)



## ----fsm_only----------------------------------------------------------------------------------------------------------------------


 ### FSM-eligible subgroup ###
  ## this is necessary because some statistics have to be computed from the analysis sample
fsm_df <- master %>%
 filter(In_Primary_analysis == 1 & fsm_admin==1)


  ## Compute results of main model
   ## Call the function - SUbstitute for the right covariates in the model
results_fsm_only <- treatment_effect_ols(
 data = fsm_df, # dataframe to be used
 y = "Overall_reading_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin", "YearGroup", "month_of_test"),
```

```
 cluster = "School_ID" # cluster variable
)

 ## Save the treatment coefficient

treatment_only_fsm <- results_fsm_only[[2]]
print(treatment_only_fsm)



## ----fsm_interaction--------------------------------------------------------------------------------------------------
 ### estimate treatment effect for FSM with an interaction term ###

 ## Compute unadjusted difference in means:
  # This is usually the difference in outcome between treatment and control
  # As this is used as a comparison by the EEF with the treatment effect, we can
  # compute a difference in differences (difference btw T and C for FSM eligible pupils
  # minus difference for non-eligible: (T-c)|FSM==1 - (T-c)|FSM==0

 mean_df <- primary_df %>%
  group_by(treatment_allocation, fsm_admin) %>%
   # Ignore missing values (we'll only use observations with non-missing values, as otherwise they are dropped from the regression)
  summarise(mean_y = mean(Overall_reading_score_el, na.rm = TRUE), .groups = "drop") %>%
    # reshape from long to wide format
  pivot_wider(names_from = treatment_allocation, values_from = mean_y)

 mean_df <- mean_df %>%
  mutate(
   unadjusted_diff = Treatment - Control
```

```
 )

 unadjusted_diff <- mean_df$Treatment - mean_df$Control

 diff_diff <- unadjusted_diff[[2]]- unadjusted_diff[[1]]

 print(diff_diff)

 ## Run OLS regression with interaction of FSM and treatment - use glm2 package

 model_fsm_interaction <- glm(Overall_reading_score_el ~ treatment_binary * fsm_admin + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin + YearGroup + month_of_test,
         data = primary_df,
         family = gaussian # Use linear regression - OLS
         )

 ## Compute Clustered SE by school ID using sandwich package

 clustered_se <- vcovCL(model_fsm_interaction, cluster = primary_df$School_ID, type = "HC1")

 ## Extract treatment coefficients and p-value from matrix

 coef_test <- coeftest(model_fsm_interaction, vcov = clustered_se)

 print(rownames(coef_test))

   ## Inspect matrix

 print("Regression results:")

 print(coef_test)

  # Extract interaction coefficient

 treatment_coef <- coef_test[2, 1]

 interaction_coef <- coef_test[13, 1]

 p_value <- coef_test[13, 4]
```

```
se_clustered <- coef_test[13, 2]

## Compute the 95% Confidence internal for the treatment coefficient
conf_int95 <- interaction_coef + c(-1.96, 1.96)*se_clustered

## Compute variance of Y by treatment group
var_df <- primary_df %>%
 group_by(treatment_allocation) %>%
 summarise(variance = var(Overall_reading_score_el, na.rm = TRUE), .groups = "drop")

head(var_df)

## Extract sample sizes
n_control <- sum(primary_df$treatment_allocation=="Control", na.rm = TRUE)
n_treatment <- sum(primary_df$treatment_allocation=="Treatment", na.rm = TRUE)

## Extract variances
var_control <- var_df %>%
 filter(var_df$treatment_allocation=="Control") %>%
 pull(variance)

var_treatment <- var_df %>%
 filter(var_df$treatment_allocation=="Treatment") %>%
 pull(variance)

## Compute pooled SD
pooled_sd <- sqrt(((n_control - 1)* var_control + (n_treatment-1)* var_treatment) / (n_control + n_treatment - 2))
```

```
## Compute hedges G

hedges_g <- interaction_coef / pooled_sd


## Compute 95% CI for hedges G

hedges_g_ci <- conf_int95 / pooled_sd


## Print results

cat("\n### Results for outcome: Overall_reading_score_el", "\n")

cat(" - N treatment: ", n_treatment, "\n")

cat(" - N control: ", n_control, "\n")

cat(" - Unadjusted difference in means: ", diff_diff, "\n")

cat(" - Interaction coefficient: ", interaction_coef, "\n")

cat(" - p-value of interaction term: ", p_value, "\n")

cat(" - SD of interaction term: ", se_clustered, "\n")

cat(" - 95% CI for interaction term: ", conf_int95, "\n")

cat(" - Variance of Overall_reading_score_el by treatment group:\n")

cat("    - Treatment: ", var_treatment, "\n")

cat("    - Control: ", var_control, "\n")

cat(" - Pooled SD: ", pooled_sd, "\n")

cat(" - Hedges G: ", hedges_g, "\n")

cat(" - 95% CI for the Hedges G: [", hedges_g_ci[1], ",", hedges_g_ci[2],"]", "\n")


## Sensitivity check: comparing the treatment effect from esitmating model only with FSM-eligible pupils, vs sum of treatment effect and interaction term


## Coefficient from subgroup sample

print(treatment_only_fsm)


## Coefficient from pooled regression + interaction term

treatment_plus_interaction <- treatment_coef + interaction_coef
```

```
 print(treatment_plus_interaction)


 difference_check <- treatment_only_fsm - treatment_plus_interaction

 print(difference_check)



## ----no_covariates----------------------------------------------------------------------------------------------------------


 ### Primary outcome analysis - EEF standard model, Equation 8 ###


primary_df_eef <- master %>%

 filter(!is.na(Overall_reading_score_el) & !is.na(treatment_allocation) & !is.na(randomisation_batch) &
!is.na(Overall_reading_score))


   ## Call the function - SUbstitute for the right covariates in the model

results_primary_analysis_eef <- treatment_effect_ols(

 data = primary_df_eef, # dataframe to be used

 y = "Overall_reading_score_el", # outcome

 treatment_variable = "treatment_allocation",

 treatment_binary = "treatment_binary",  # treatment in binary format for regression

   # List of covariates:

 covariates = c("randomisation_batch", "Overall_reading_score"), # we only include the stratification variable and baseline
attainment

 cluster = "School_ID" # cluster variable

)


 ### Primary outcome analysis - only treatment as covariate ###


primary_df_nocovs <- master %>%

 filter(!is.na(Overall_reading_score_el) & !is.na(treatment_allocation))
```

```
  ## Call the function - SUbstitute for the right covariates in the model
results_primary_analysis_nocovs <- treatment_effect_ols(
 data = primary_df_nocovs, # dataframe to be used
 y = "Overall_reading_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = NULL, # no covariates
 cluster = "School_ID" # cluster variable
)


 ### Secondary outcome: PC ###
df_nocovs <- master %>%
 filter(!is.na(PC_score_el) & !is.na(treatment_allocation))

  ## Call the function - SUbstitute for the right covariates in the model
results_PC_nocovs <- treatment_effect_ols(
 data = df_nocovs, # dataframe to be used
 y = "PC_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = NULL, # no covariates
 cluster = "School_ID" # cluster variable
)

 ### Secondary outcome: SC ###
```

```
df_nocovs <- master %>%
  filter(!is.na(SC_score_el) & !is.na(treatment_allocation))


    ## Call the function - SUbstitute for the right covariates in the model
results_SC_nocovs <- treatment_effect_ols(
  data = df_nocovs, # dataframe to be used
  y = "SC_score_el", # outcome
  treatment_variable = "treatment_allocation",
  treatment_binary = "treatment_binary",  # treatment in binary format for regression
    # List of covariates:
  covariates = NULL, # no covariates
  cluster = "School_ID" # cluster variable
)


  ### Secondary outcome: KS2 reading ###
df_nocovs <- master %>%
  filter(!is.na(KS2_READSCORE_num) & !is.na(treatment_allocation))


results_KS2_nocovs <- treatment_effect_ols(
  data = df_nocovs, # dataframe to be used
  y = "KS2_READSCORE_num", # outcome
  treatment_variable = "treatment_allocation",
  treatment_binary = "treatment_binary",  # treatment in binary format for regression
    # List of covariates:
  covariates = NULL, # we only include the stratification variable
  cluster = "School_ID" # cluster variable
)
```

Appendix M: Balance checks code

```
## ----setup----------------------------------------------------------------------------------------------------------------

#DELETED FOR CLEARANCE



## ----setup2---------------------------------------------------------------------------------------------------------------
  ### Set up ###

## Load packages
library(tidyverse)
library(rmarkdown)
library(ggplot2)

## Check current directory - It should be the same folder of the quarto document
getwd()



## ----load_data------------------------------------------------------------------------------------------------------------

## Clean current working environment
rm(list = ls())

## Load main dataset
  ## THIS CHUNK ONLY RUNS PROPERLY IN QUARTO"SOURCE" MODE (AS OPPOSED TO VISUAL MODE)
#DELETED FOR CLEARANCE

master <- master %>%
```

```
 select(-...1)



## ----continuous_vars----------------------------------------------------------------------------------------------------
## Set up the function
normalised_diff_continuous <- function(data, continuous_var) {


 stats_continuous_var <- data %>%
  group_by(treatment_allocation) %>% # Get results by trial arm
  summarise(
   count = sum(!is.na(!!sym(continuous_var))), # Number of observations
   missing = sum(is.na(!!sym(continuous_var))), # Missing observations
   mean = mean(!!sym(continuous_var), na.rm = TRUE), # Mean
   sd = sd(!!sym(continuous_var), na.rm = TRUE) # sd
  )



 # Compute normalised differences
 norm_diff_continuous_var <- stats_continuous_var %>%
  summarise(
   pooled_SD = sqrt(0.5*(sd[treatment_allocation=="Treatment"]^2+sd[treatment_allocation=="Control"]^2)) ,
    # Normalised difference = difference in mean / pooled SD
   norm_diff = (mean[treatment_allocation=="Treatment"] - mean[treatment_allocation=="Control"]) /pooled_SD,
    # Pooled SD using small sample correction for Hedges G
   pooled_SD_HG = sqrt(((count[treatment_allocation=="Treatment"]-
1)*sd[treatment_allocation=="Treatment"]^2+(count[treatment_allocation=="Control"]-
1)*sd[treatment_allocation=="Control"]^2)/(count[treatment_allocation=="Treatment"]+count[treatment_allocation=="Contr
ol"]-2)),
    # Effect size in Hedges g = difference in mean / pooled SD with correction
   norm_diff_HG = (mean[treatment_allocation=="Treatment"] - mean[treatment_allocation=="Control"]) /pooled_SD_HG
```

```
  )
 # Store results in a list
 return(list(
  summary = stats_continuous_var,
  norm_diff = norm_diff_continuous_var
 ))
}



## ----continuous_vars_results----------------------------------------------------------------------------------------------------

 ### Baseline NGRT score ###
results_baseline_NGRT_overall <- normalised_diff_continuous(
 data = master,
 continuous_var = "Overall_reading_score"
)
print("Results for baseline NGRT overall score")
print(results_baseline_NGRT_overall$summary)
print(results_baseline_NGRT_overall$norm_diff)

 ### Baseline NGRT Sentence Completion ###
results_baseline_NGRT_SC <- normalised_diff_continuous(
 data = master,
 continuous_var = "SC_score"
)
print("Results for baseline NGRT Sentence Completion score")
print(results_baseline_NGRT_SC$summary)
print(results_baseline_NGRT_SC$norm_diff)
```

```
 ### Baseline NGRT Passage Comprehension ###
results_baseline_NGRT_PC <- normalised_diff_continuous(
 data = master,
 continuous_var = "PC_score"
)
print("Results for baseline NGRT Passage Comprehension score")
print(results_baseline_NGRT_PC$summary)
print(results_baseline_NGRT_PC$norm_diff)

 ### Month of test ###
 ## This is not a baseline pupil characteristic but we need to know
 ## if month of test is balanced or not (to see if control and treatment pupils did the test roughly at the same time)

  # First: transform month of test to numeric
master <- master %>%
 mutate(
  month_of_test_factor = factor(month_of_test,
                                levels = c("March", "April", "May", "June", "July")),
  month_num = case_when(
   month_of_test == "March" ~ 3,
   month_of_test == "April" ~ 4,
   month_of_test == "May" ~ 5,
   month_of_test == "June" ~ 6,
   month_of_test == "July" ~ 7,
   NA ~ NA,
   TRUE ~ NA
  )
 )
```

```
 # Do a first visual inspection
ggplot(master, aes( x = month_of_test_factor)) +
 geom_bar(fill = "steelblue", color = "black") +
 labs(title  ="Frequency of test months",
     x = "Month",
     y = "Count") +
 theme_minimal()

 # Get results for month of test
results_month_test <- normalised_diff_continuous(
 data = master,
 continuous_var = "month_num"
)
print("Results for month of test")
print(results_month_test$summary)
print(results_month_test$norm_diff)


## ----binary_vars----------------------------------------------------------------------------------------------------
## Set up the function

normalised_diff_binary <- function(data, binary_var) {

 stats_binary_var <- data %>%
  group_by(treatment_allocation) %>% # Get results by trial arm
 summarise(
  n = n(), # TOtal number of observations
  count = sum(!is.na(!!sym(binary_var))), # Number of non-missing observations
```

```
  missing = sum(is.na(!!sym(binary_var))), # Missing
  count_1 = sum(!!sym(binary_var)==1, na.rm = TRUE),
  count_0 = sum(!!sym(binary_var)==0, na.rm = TRUE),
  percent_1 = mean(!!sym(binary_var), na.rm = TRUE), # Proportion of 1
  percent_0 = 1-mean(!!sym(binary_var), na.rm = TRUE), # Proportion of 0
  sd = sd(!!sym(binary_var), na.rm = TRUE)  )

 print(stats_binary_var)

 # Compute normalised differences
 norm_diff_binary_var <- stats_binary_var %>%
  summarise(
   pooled_SD = sqrt(0.5*(sd[treatment_allocation=="Treatment"]^2+sd[treatment_allocation=="Control"]^2)) ,
   norm_diff = (percent_1[treatment_allocation=="Treatment"] - percent_1[treatment_allocation=="Control"]) /pooled_SD ,
       # Pooled SD using small sample correction for Hedges G
   pooled_SD_HG = sqrt(((count[treatment_allocation=="Treatment"]-
1)*sd[treatment_allocation=="Treatment"]^2+(count[treatment_allocation=="Control"]-
1)*sd[treatment_allocation=="Control"]^2)/(count[treatment_allocation=="Treatment"]+count[treatment_allocation=="Contr
ol"]-2)),
     # Effect size in Hedges g = difference in mean / pooled SD with correction
   norm_diff_HG = (percent_1[treatment_allocation=="Treatment"] - percent_1[treatment_allocation=="Control"])
/pooled_SD_HG
  )

 print(norm_diff_binary_var)

 # Store results in a list
 return(list(
  summary = stats_binary_var,
  norm_diff = norm_diff_binary_var
 ))
```

```
}



## ----binary_vars_results--------------------------------------------------------------------------------------------------
 ### FSM status ###
results_fsm <- normalised_diff_binary(
 data = master,
 binary_var = "fsm_admin"
)
print("Results for FSM admin")
print(results_fsm$summary)
print(results_fsm$norm_diff)

 ### EAL status ###
results_eal <- normalised_diff_binary(
 data = master,
 binary_var = "eal_admin"
)
print("Results for EAL status admin")
print(results_eal$summary)
print(results_eal$norm_diff)

 ### Gender ###
master %>% count(gender_admin)
master <- master %>%
 mutate(
  gender_admin_num = case_when(
   gender_admin == "Female" ~ 1,
```

```
    gender_admin == "Male" ~ 0,
    TRUE ~ NA
  )
 )

# LB QA check inspect gender variable
table(master$gender_admin)
table(master$Gender) # OK.

master %>% count(gender_admin, gender_admin_num)

results_gender <- normalised_diff_binary(
 data = master,
 binary_var = "gender_admin_num"
)
print("Results for gender admin")
print(results_gender$summary)
print(results_gender$norm_diff)

 ### Year 5 vs 6 ###
  # as there was only 5 and 6, we can use a binary variable
master %>% count(YearGroup)
master <- master %>%
 mutate(
  year5 = if_else(
   YearGroup == 5, 1,0
  )
 )
master %>% count(year5, YearGroup)
```

```
results_yeargroup <- normalised_diff_binary(
  data = master,
  binary_var = "year5"
)
print("Results for Year Group")
print(results_yeargroup$summary)
print(results_yeargroup$norm_diff)



## ----pupil_analysis----------------------------------------------------------------------------------------------------
primary_df <- master %>%
  filter(In_Primary_analysis==1)



### Baseline NGRT score ###
results_baseline_NGRT_overall_analysis <- normalised_diff_continuous(
  data = primary_df,
  continuous_var = "Overall_reading_score"
)
print("Results for baseline NGRT overall score, at analysis")
print(results_baseline_NGRT_overall_analysis$summary)
print(results_baseline_NGRT_overall_analysis$norm_diff)

  ### Baseline NGRT Sentence Completion ###
results_baseline_NGRT_SC_analysis <- normalised_diff_continuous(
  data = primary_df,
  continuous_var = "SC_score"
```

```
)
print("Results for baseline NGRT Sentence Completion score, at analysis")
print(results_baseline_NGRT_SC_analysis$summary)
print(results_baseline_NGRT_SC_analysis$norm_diff)

 ### Baseline NGRT Passage Comprehension ###
results_baseline_NGRT_PC_analysis <- normalised_diff_continuous(
 data = primary_df,
 continuous_var = "PC_score"
)
print("Results for baseline NGRT Passage Comprehension score, at analysis")
print(results_baseline_NGRT_PC_analysis$summary)
print(results_baseline_NGRT_PC_analysis$norm_diff)

 ### Month of test ###
results_month_test_analysis <- normalised_diff_continuous(
 data = primary_df,
 continuous_var = "month_num"
)
print("Results for month of test at analysis")
print(results_month_test_analysis$summary)
print(results_month_test_analysis$norm_diff)

 ### FSM status ###
results_fsm_analysis <- normalised_diff_binary(
 data = primary_df,
 binary_var = "fsm_admin"
)
print("Results for FSM admin at analysis")
```

```
print(results_fsm_analysis$summary)
print(results_fsm_analysis$norm_diff)

 ### EAL status ###
results_eal_analysis <- normalised_diff_binary(
 data = primary_df,
 binary_var = "eal_admin"
)
print("Results for EAL status admin at analysis")
print(results_eal_analysis$summary)
print(results_eal_analysis$norm_diff)

 ### Gender ###
results_gender_analysis <- normalised_diff_binary(
 data = primary_df,
 binary_var = "gender_admin_num"
)
print("Results for gender admin at analysis")
print(results_gender_analysis$summary)
print(results_gender_analysis$norm_diff)

 ### Year 5 vs 6 ###

results_yeargroup_analysis <- normalised_diff_binary(
 data = primary_df,
 binary_var = "year5"
)
print("Results for Year Group at analysis")
print(results_yeargroup_analysis$summary)
```

```
print(results_yeargroup_analysis$norm_diff)



cat("LB QA check")
normalised_difference <- function(df, covar.col) {

 # Get treatment mean
 mean_treatment <- df %>%
  filter(treatment_allocation=="Treatment") %>%
  pull({{covar.col}}) %>%
  mean(., na.rm = TRUE)

 # Store treatment mean
 mean_treatment <- mean_treatment[[1]]

 # Get treatment sd
 sd_treatment <- df %>%
  filter(treatment_allocation=="Treatment") %>%
  pull({{covar.col}}) %>%
  sd(., na.rm = TRUE)

 # Store treatment sd
 sd_treatment <- sd_treatment[[1]]

  # Get control mean
 mean_control <- df %>%
  filter(treatment_allocation=="Control") %>%
  pull({{covar.col}}) %>%
  mean(., na.rm = TRUE)
```

```
 # Store control mean
 mean_control <- mean_control[[1]]

 # Get control sd
 sd_control <- df %>%
  filter(treatment_allocation=="Control") %>%
  pull({{covar.col}}) %>%
  sd(., na.rm = TRUE)

 # Store control sd
 sd_control <- sd_control[[1]]

 # Compute normalised difference
 norm_diff <- (mean_treatment - mean_control)/sqrt(0.5*(sd_treatment^2 + sd_control^2))

 # Output
 output <- tibble("Covariate" = covar.col,
          "Treatment mean" = round(mean_treatment, digits = 5),
          "Treatment sd" = round(sd_treatment, digits = 5),
          "Control mean" = round(mean_control, digits = 5),
          "Control sd" = round(sd_control, digits = 5),
          "Normalised difference" = round(norm_diff, digits = 5))

 output
}

# compute normalised difference and compare with NT results for one covariate
normalised_difference(master, "gender_admin_num")
```

```
print(results_gender$summary)

print(results_gender$norm_diff)


normalised_difference(primary_df, "gender_admin_num")

print(results_gender_analysis$summary)

print(results_gender_analysis$norm_diff)
```

Appendix N: ICC and power calculations code

```
## ----setup2----------------------------------------------------------------------------------------------------------

  ### Set up ###


## Load packages

library(tidyverse)

library(rmarkdown)

library(ggplot2)

library(pwr)

library(glm2)

library(lmtest)

library(lme4)


## Check current directory - It should be the same folder of the quarto document

getwd()




## ----load_data-------------------------------------------------------------------------------------------------------


## Clean current working environment

rm(list = ls())


## Load main dataset

  ## THIS CHUNK ONLY RUNS PROPERLY IN QUARTO"SOURCE" MODE (AS OPPOSED TO VISUAL MODE)

#DELETED FOR CLEARANCE


master <- master %>%

 select(-...1)
```

```
## ----power_calcs----------------------------------------------------------------------------------------------------
  # 1. Filter only observations at analysis
df_analysis <- master %>%
 filter(In_Primary_analysis==1)

 # Report number of pupils and number of schools, full sample
cat("Number of pupils at analysis, by treatment arm:")
table(df_analysis$treatment_allocation)

cat("Number of schools at analysis, by treatment arm:")
schools <- df_analysis %>%
 group_by(treatment_allocation) %>%
 summarise(
  total_schools = n_distinct(School_ID)
 )
print(schools)

 # Compare to total number of schools at randomisation:
schools2 <- master %>%
 group_by(treatment_allocation) %>%
 summarise(
  total_schools = n_distinct(School_ID)
 )

# Get FSM subsample dataframe
```

```
df_analysis_fsm <- master %>%
 filter(In_Primary_analysis==1 & fsm_admin==1)

# Report number of pupils and number of schools, FSM subsample
cat("Number of pupils at analysis, FSM subsample, by treatment arm:")
table(df_analysis_fsm$treatment_allocation)

cat("Number of schools at analysis, FSM subsample, by treatment arm:")
schools_fsm <- df_analysis_fsm %>%
 group_by(treatment_allocation) %>%
 summarise(
  total_schools = n_distinct(School_ID)
 )
print(schools_fsm)

 # 2. Compute cluster size as the number of pupils in each school, and
 # FSm cluster size as number of FSM pupils in each school

df_cluster_size <- df_analysis %>%
 group_by(School_ID) %>%
 summarise(cluster_size = n(),
        cluster_size_FSM = sum(fsm_admin == 1),
        Year5 = sum(YearGroup=="5"),
        Year6 = sum(YearGroup=="6"),
        treatment_allocation = first(treatment_allocation))

 # 3.Set power assumptions for all sample power calcs
  # 3.1 Set average cluster size at analysis
average_cluster_size <- mean(df_cluster_size$cluster_size)
```

```
print(paste0("Average cluster size = ", average_cluster_size))


average_cluster_size_fsm <- mean(df_cluster_size$cluster_size_FSM)
print(paste0("Average FSM cluster size = ", average_cluster_size_fsm))


  # 3.2 Set number of clusters at analysis
num_clusters <- length(unique(df_cluster_size$School_ID))


  # 3.3 Compute baseline correlation
  # First: transform month of test to numeric
df_analysis <- df_analysis %>%
 mutate(
  month_num = case_when(
   month_of_test == "March" ~ 3,
   month_of_test == "April" ~ 4,
   month_of_test == "May" ~ 5,
   month_of_test == "June" ~ 6,
   month_of_test == "July" ~ 7,
   NA ~ NA,
   TRUE ~ NA
  )
 )


# LB QA check
table(df_analysis$month_of_test) # OK.
model <- glm2(Overall_reading_score_el ~ Overall_reading_score + gender_admin + eal_admin + YearGroup,
          data = df_analysis,
          family = gaussian)
```

```
model_summary <- summary(model)
print(model_summary)

    # Obtain R squared for overall sample from output
deviance <- summary(model)$deviance
null_deviance <- summary(model)$null.deviance
r2_overall <- 1 - (deviance/null_deviance)

r2_alternative <- cor(predict(model), model$y)^2 # gives exactly the same.
cat("\n### Estimated correlation between outcome and baseline covariates (full sample): \n")
cat(" - Way 1:",r2_overall, " \n")
cat(" - Way 2:",r2_alternative, " \n")

 ## Re-estimate but only rpre test and post test  NGRT scores
model2 <- glm2(Overall_reading_score_el ~ Overall_reading_score,
          data = df_analysis,
          family = gaussian)

model_summary <- summary(model)
print(model_summary)

    # Obtain R squared for overall sample from output
deviance <- summary(model2)$deviance
null_deviance <- summary(model2)$null.deviance
r2_overall2 <- 1 - (deviance/null_deviance)
cat("\n### Estimated correlation between pre-test and post test in NGRT overall reading score (full sample): \n")
cat(" - Way 1:",r2_overall2, " \n")

    # Regress outcome on baseline covariates - FSM sample
```

```
model_fsm <- glm2(Overall_reading_score_el ~ Overall_reading_score + gender_admin + eal_admin + YearGroup,
          data = df_analysis_fsm,
          family = gaussian)

model_summary <- summary(model_fsm)
print(model_summary)

    # Obtain R squared for overall sample from output
deviance <- summary(model_fsm)$deviance
null_deviance <- summary(model_fsm)$null.deviance
r2_fsm <- 1 - (deviance/null_deviance)

cat("\n### Estimated correlation between outcome and baseline covariates (FSM sample):", r2_fsm, "\n")



  # 3.4  Compute ICC - I'm using the lme4 package. I can't find ICC in the R package repository.

     # STep 1:  fit the random model with school as factor
model_icc <- lmer(Overall_reading_score_el ~
              1 +
              (1|School_ID),        # Random intercept for each school
            data = df_analysis)

var_components <- VarCorr(model_icc)
print(var_components)

     # Step 2: extract school level variance
school_var <- as.numeric(var_components$School_ID)
print(school_var)
```

```
    # Step 3: extract residual variance by squaring the residual SD
residual_var <- attr(var_components, "sc")^2 # VarCorr stores the residual SD as sc
print(residual_var)

    # Step 4: Compute ICC
icc_overall <- school_var / (school_var + residual_var)
print(paste0("ICC at the school level = ", icc_overall))

   # Repeat for FSM
model_icc_fsm <- lmer(Overall_reading_score_el ~
             1 +              # Add 1 intercept
             (1|School_ID),       # Random intercept for each school
           data = df_analysis_fsm)
var_components_fsm <- VarCorr(model_icc_fsm)
print(var_components_fsm)

school_var_fsm <- as.numeric(var_components_fsm$School_ID)
print(school_var_fsm)

residual_var_fsm <- attr(var_components_fsm, "sc")^2 # VarCorr stores the residual SD as sc
print(residual_var_fsm)

icc_fsm <- school_var_fsm / (school_var_fsm + residual_var_fsm)
print(paste0("ICC at the school level, FSM subsample = ", icc_fsm)) # LB QA: reviewed, all OK.

# LB QA tip slightly shorter code:
# school_var_fsm <- as.numeric(VarCorr(model_fsm)$School_ID)
# residual_var_fsm <- attr(VarCorr(model_fsm), "sc")^2
```

```
# icc_fsm <- school_var_fsm / (school_var_fsm + residual_var_fsm)



  # 3.5 Set assumptions on alpha an power
sig_level <- 0.05
power <- 0.8



 # 4. Use the ICC to compute the effective sample size for all clusters
  # ESS = Cluster_size / (1 + (cluster_size - 1)* ICC)

df_cluster_size <- df_cluster_size %>%
 mutate(
  ess_overall = cluster_size / (1 + (cluster_size - 1)*icc_overall) ,
  ess_fsm = cluster_size_FSM/ (1 + (cluster_size_FSM - 1)*icc_fsm)
 ) # LB QA: reviewed, OK.

 # Sum up all ESS for treatment and control to get overall sample
ess_overall_control <- round(
 sum(df_cluster_size$ess_overall[df_cluster_size$treatment_allocation == "Control"]), 0)

ess_overall_treatment <- round(
 sum(df_cluster_size$ess_overall[df_cluster_size$treatment_allocation == "Treatment"]), 0)

ess_fsm_control <- round(
 sum(df_cluster_size$ess_fsm[df_cluster_size$treatment_allocation == "Control"]), 0)

ess_fsm_treatment <- round(
 sum(df_cluster_size$ess_fsm[df_cluster_size$treatment_allocation == "Treatment"]), 0)
```

```
 # 5. COmpute MDES using power.t2n.test (bc of difference ins ample sizes)  and ess


  # ALl sample #
mdes_overall <- pwr.t2n.test(n1 = ess_overall_treatment,
                    n2 = ess_overall_control,
                    d = NULL,
                    sig.level = sig_level,
                    power = power,
                    alternative = c("two.sided"))
print(mdes_overall)

mdes_overall_d <- mdes_overall$d # Extract Cohens D
print(mdes_overall_d)

# Adjust for covariates
mdes_overall_d_adjusted <- mdes_overall_d*sqrt(1 - r2_overall)
print(mdes_overall_d_adjusted)

# Transform to Hedges G
hedges_g <- function(n_treatment, n_control, cohens_d){
 df <- n_treatment + n_control - 2
 cohens_d*(1-(3/(4*df-1)))
}

mdes_overall_hedges_g <- hedges_g(ess_overall_treatment, ess_overall_control, mdes_overall_d_adjusted)
print(paste0("MDES at analysis = ", mdes_overall_hedges_g))
```

```
  # FSM subsample #
mdes_fsm <- pwr.t2n.test(n1 = ess_fsm_treatment,
                         n2 = ess_fsm_control,
                         d = NULL,
                         sig.level = sig_level,
                         power = power,
                         alternative = c("two.sided"))
print(mdes_fsm)

mdes_fsm_d <- mdes_fsm$d # Extract Cohens D
print(mdes_fsm_d)

# Adjust for covariates
mdes_fsm_d_adjusted <- mdes_fsm_d*sqrt(1 - r2_fsm)
print(mdes_fsm_d_adjusted)

mdes_fsm_hedges_g <- hedges_g(ess_fsm_treatment, ess_fsm_control,mdes_fsm_d_adjusted )

print(paste0("MDES at analaysis for FSM = ", mdes_fsm_hedges_g))



## ----icc_ngrt----------------------------------------------------------------------------------------------------------------

 # 1. ICC at pre-test
model_icc_pretest <- lmer(Overall_reading_score ~
              1 +               # Add 1 intercept
              (1|School_ID),        # Random intercept for each school
            data = master)
```

```
var_components <- VarCorr(model_icc_pretest)
print(var_components)

    # Step 2: extract school level variance
school_var <- as.numeric(var_components$School_ID)
print(school_var)

    # Step 3: extract residual variance by squaring the residual SD
residual_var <- attr(var_components, "sc")^2 # VarCorr stores the residual SD as sc
print(residual_var)

    # Step 4: Compute ICC
icc_overall_pretest <- school_var / (school_var + residual_var)
print(paste0("ICC at the school level, pre-test = ", icc_overall_pretest))

 print(paste0("ICC at the school level, post-test = ", icc_overall))


## ----icc_ks2--------------------------------------------------------------------------------------------------------------
 # Filter observations used in the KS2 model
ks2_df <- master %>%
 filter(!is.na(treatment_allocation) &  !is.na(randomisation_batch) &
    !is.na(KS2_READSCORE_num) & !is.na(Overall_reading_score))

model_icc_ks2 <- lmer(KS2_READSCORE_num ~
              1 +              # Add 1 intercept
              (1|School_ID),       # Random intercept for each school
            data = ks2_df)
```

```
var_components <- VarCorr(model_icc_ks2)
print(var_components)

    # Step 2: extract school level variance
school_var <- as.numeric(var_components$School_ID)
print(school_var)

    # Step 3: extract residual variance by squaring the residual SD
residual_var <- attr(var_components, "sc")^2 # VarCorr stores the residual SD as sc
print(residual_var)

    # Step 4: Compute ICC
icc_overall_ks2 <- school_var / (school_var + residual_var)
print(paste0("ICC at the school level, KS2 reading scores = ", icc_overall_ks2))

# LB re-QA check - using instead dataset with non-missing KS2 score and no other restriction on covariates
ks2_df2 <- master %>%
 filter(!is.na(KS2_READSCORE_num))

model_icc_ks2_2 <- lmer(KS2_READSCORE_num ~
              1 +              # Add 1 intercept
              (1|School_ID),       # Random intercept for each school
            data = ks2_df2)

var_components <- VarCorr(model_icc_ks2_2)
print(var_components)

    # Step 2: extract school level variance
school_var <- as.numeric(var_components$School_ID)
```

```
print(school_var)

    # Step 3: extract residual variance by squaring the residual SD
residual_var <- attr(var_components, "sc")^2 # VarCorr stores the residual SD as sc
print(residual_var)

    # Step 4: Compute ICC
icc_overall_ks2 <- school_var / (school_var + residual_var)
print(paste0("ICC at the school level, KS2 reading scores = ", icc_overall_ks2))
```

Appendix O: Participant flow and attrition code

```
## ----load_data-----------------------------------------------------------------------------------------------------------

## Load packages

library(tidyverse)

library(stringdist)

library(lubridate)

library(rmarkdown)

## Clean current working environment

rm(list = ls())

## Load main dataset

   ## THIS CHUNK ONLY RUNS PROPERLY IN QUARTO"SOURCE" MODE (AS OPPOSED TO VISUAL MODE)

#DELETED FOR CLEARANCE

master <- master %>%

  select(-...1)



## ----diagram-------------------------------------------------------------------------------------------------------------

 # Allocation

cat("Number of pupils randomised:")

table(master$treatment_allocation)

master <- master %>%

  mutate(ngrt_missing = is.na(Overall_reading_score_el))

cat("Number of pupils lost to follow up:")
```

```
table(master$ngrt_missing, master$treatment_allocation)

cat("Number of pupils with outcome data collected:")

master_outcome <- master %>%

 filter(!is.na(Overall_reading_score_el))

table(master_outcome$treatment_allocation)


cat("Number of pupils analysed:")

 # In primary analysis

table(master_outcome$treatment_allocation, master_outcome$In_Primary_analysis)




## ----attrition----------------------------------------------------------------------------------------------------------

 # Attrition

attrition <- master %>%

 group_by(In_Primary_analysis, treatment_allocation) %>%

 summarise(

  count = n(), .groups = "drop"

   )


attrition <- master %>%

 count(In_Primary_analysis, treatment_allocation) %>%

  group_by(treatment_allocation) %>%

 mutate(

  att_rate = round( n / sum(n)*100, 2)

 )

cat("Attrition by treatment arm: ")

print(attrition)
```

```
attrition_total <- master %>%
 count(In_Primary_analysis) %>%
 mutate(
  att_rate = round( n / sum(n)*100, 2)
 )
cat("Attrition rate, total: ")
print(attrition_total)



## ----attrition_reasons----------------------------------------------------------------------------------------------------


 # Their parents withdrew them from evaluation:
# LB QA check
cat("Parents withdrew their consent:")
table(master$no_parental_consent)


master <- master %>%
 mutate(
  no_parental_consent = if_else(is.na(no_parental_consent), 0, no_parental_consent)
 )
table(master$no_parental_consent)


 # Clean attrition variables
table(master$attrition)


master <- master %>%
 mutate(
  attrition = if_else(is.na(attrition), 0, attrition)
 )
```

```
cat("Pupils are not analysed for other reasons:")
table(master$attrition)

table(master$no_parental_consent, master$attrition) # there is no Overlap between variables
table(master$attrition)
 # Variable attrition_notes includes the notes on reasons for attrition collected from the schools
table(master$attrition_notes)
# LB QA check
length(unique(master$attrition_notes))

 # Clean the variable
master <- master %>%
  mutate(att_notes = attrition_notes)

master <- master %>%
 mutate(
  att_notes = case_when(
    grepl("No longer", attrition_notes) ~ "Pupil has left the school",
    grepl("Opted out", attrition_notes) ~ "Opted out by teacher/the school",
    grepl("Removed from", attrition_notes) ~ "Opted out by teacher/the school",
    grepl("Longstanding illness", attrition_notes) ~ "Persistent absence",
    grepl("Withdrawn", attrition_notes) ~ "Opted out by teacher/the school",
    grepl("Withdrawn", attrition_notes) ~ "Opted out by teacher/the school", # QA: repetition?
    grepl("Withdrew", attrition_notes) ~ "Pupil didn't assent to testing",
    grepl("withdrew consent", attrition_notes) ~ "Pupil didn't assent to testing",
    grepl("withdrew their consent", attrition_notes) ~ "Pupil didn't assent to testing",
    grepl("Refused to complete", attrition_notes) ~ "Pupil didn't assent to testing",
    grepl("Absent since Easter, unlikely to return on time", attrition_notes) ~ "Persistent absence",
     grepl("Won't be back to school", attrition_notes) ~ "Persistent absence",
```

```
    grepl("often off school", attrition_notes) ~ "Persistent absence",
    TRUE ~ att_notes
   )
 )
table(master$att_notes)

 # Check all observations have been cleaned
master <- master %>%
 mutate(
  attrition_notes_1 = !is.na(attrition_notes),
  att_notes_2 = !is.na(att_notes)
 )
table(master$attrition_notes_1, master$att_notes_2) # All clean
master <- master %>%
 select(-c("attrition_notes_1", "att_notes_2"))

table(master$att_notes, master$ngrt_missing, useNA = "ifany")

master <- master %>%
 mutate(
  att_notes = if_else(
   ngrt_missing == FALSE, NA, att_notes
  )
 )
table(master$att_notes, master$ngrt_missing, useNA = "ifany")

table(master$att_notes, master$In_Primary_analysis, useNA = "ifany")

master <- master %>%
```

```
 mutate(
  att_notes = if_else(
   no_parental_consent == 0 & In_Primary_analysis == 0 & ngrt_missing == TRUE & is.na(att_notes), "Reason not known",
att_notes
  )
 )
table(master$att_notes, master$ngrt_missing, useNA = "ifany")


master <- master %>%
 mutate(
  att_notes = if_else(
   no_parental_consent == 1, "Withdrawn from trial by parents"
  , att_notes),
  att_notes = if_else(
   no_parental_consent == 0 & In_Primary_analysis == 0 & ngrt_missing == FALSE , "Missing covariates", att_notes
  )
 )
table(master$att_notes, master$In_Primary_analysis, useNA = "ifany")


  ## Get the counts and percentages for the different reasons for attrition ##
attrited_pupils <- master %>%
 filter(In_Primary_analysis==0)


attrition_notes <- attrited_pupils %>%
 count(att_notes) %>%
 mutate(
  att_notes_perc = round( n / sum(n)*100, 2)
 )
cat("Other reasons for attrition")
print(attrition_notes)
```

Appendix P: CACE analysis and dosage code

```
## ----setup2, echo = FALSE, results='hide'--------------------------------------------------------------------------------

## Load packages
library(tidyverse)
library(glm2) # for all regression models if needed
library(broom)
library(ggplot2)
library(sandwich) # for clustered standard errors
library(zoo)
library(lmtest)
library(rmarkdown)
library(AER) # For IV regressions

## Clean current working environment
rm(list = ls())

## Check current directory - It should be the same folder of the quarto document
getwd()



## ----load_data, echo = FALSE, results='hide'-----------------------------------------------------------------------------
## Load main dataset
#DELETED FOR CLEARANCE

master <- master %>%
  select(-...1)
```

```
## ----cleaning----------------------------------------------------------------------------------------------------------

## Transform categorical covariates into numeric

 # Transform gender_admin into a binary variable
master <- master %>%
 mutate(
  gender_admin_num = if_else(gender_admin == "Female", 1, 0)
 )
master %>% count(gender_admin, gender_admin_num)

 # Transform month of test to numeric
master <- master %>%
 mutate(
  month_of_test_factor = factor(month_of_test,
                                levels = c("March", "April", "May", "June", "July")),
  month_num = case_when(
   month_of_test == "March" ~ 3,
   month_of_test == "April" ~ 4,
   month_of_test == "May" ~ 5,
   month_of_test == "June" ~ 6,
   month_of_test == "July" ~ 7,
   NA ~ NA,
   TRUE ~ NA
  )
 )
master %>% count(month_of_test, month_of_test_factor)
```

```
 # Transform treatment variable into factor ###
master <- master %>%
 mutate(
  treatment_binary = if_else(treatment_allocation=="Treatment", 1, 0)
 ) %>%
 relocate(treatment_binary, .after = treatment_allocation)

master %>% count(treatment_allocation, treatment_binary)



## ----CACE_function, echo = FALSE, results='hide'-----------------------------------------------------------------------
 ### Setting up the function ###

 ## CACE model: compliance = #DELETED FOR CLEARANCE sessions. ##
  ### Treatment assignment is the instrument

 ## Ivreg command doesn't give the results of both stages, so they will have to
 ## be computed separately # LB QA:

CACE_model <- function(data, y, treatment_variable, treatment_binary, compliance_variable, covariates = NULL, cluster){
  mean_df <- data %>%
  group_by(!!sym(treatment_variable)) %>%
   # Ignore missing values (we'll only use observations with non-missing values, as otherwise they are dropped from the
regression)
  summarise(mean_y = mean(!!sym(y), na.rm = TRUE), .groups = "drop") %>%
   # reshape from long to wide format
  pivot_wider(names_from = !!sym(treatment_variable), values_from = mean_y)

 unadjusted_diff <- mean_df$Treatment - mean_df$Control
```

```
## Set regression formula for the first stage
stage1_regression_formula <- as.formula(
  paste(compliance_variable, "~", treatment_binary,
        if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else "")
)

  ## Set regression formulas for ivreg command:
     ## First stage for ivreg - it doesn't include the endogenous variable that needs the instrument
stage1_ivreg_formula <- paste(treatment_binary,
        if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else "")



print(stage1_ivreg_formula)

    ## Second stage for ivreg
stage2_ivreg_formula <- as.formula(
  paste(y, "~", compliance_variable,
        if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else "", "|", treatment_binary,
        if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else ""))

print(stage2_ivreg_formula)

## Run the 1st stage independently
first_stage <- lm(stage1_regression_formula,
             data = data)

## Compute Clustered SE by school ID using sandwich package
  # Clustered SE
```

```
first_stage_se <- coeftest(first_stage,
                           vcov = vcovCL,
                           cluster = data[[cluster]],
                           data = data)

## Print results of first stage
cat("*** First stage results: *** \n")

cat(" Equation:\n ")
print(stage1_regression_formula)

cat(" Results:\n ")
cat(first_stage_se)

print(first_stage_se)

# First stage F statistic - smaller than 10 indicates a weak instrument as rule of thumb

first_stage_summary <- summary(first_stage)

f_stat <- first_stage_summary$fstatistic
print(f_stat)

p_value <- pf(f_stat[1], f_stat[2], f_stat[3], lower.tail = FALSE)

cat("*** First stage F-statistic (not clustered - ignore): ***\n")

cat("F(", f_stat[2], ",", f_stat[3], ") = ", round(f_stat[1], 2),
   ", p-value: ", format.pval(p_value), "\n", sep="")
```

```
## Laure's QA way:
vcov_clustered <- vcovCL(first_stage,
                         cluster = data[[cluster]])

ftest <- waldtest(first_stage, vcov = vcov_clustered, test = "F")
f_stat2 <- ftest[, "F"]
p_value2 <-  ftest[, "Pr(>F)"]
df <-  ftest[, "Df"]
resdf <- ftest[, "Res.Df"]

cat("\n### *** First stage F-statistic - clustered version: ***\n")
cat("F(", df[2], ",", resdf[1], ") = ", round(f_stat2[2], 2),
   ", p-value: ", format.pval(p_value2), "\n", sep="")

 ## 2nd stage
second_stage <- ivreg(stage2_ivreg_formula,
                      data = data,
                      x = TRUE)
## Compute Clustered SE by school ID using sandwich package
coef_test <- coeftest(second_stage,
                        vcov = vcovCL,
                        cluster = data[[cluster]],
                        data = data)

cat("\n### *** Second stage results: ***\n")
coef_test

## Extract treatment coefficients and p-value from matrix
```

```
print(rownames(coef_test))


  ## Inspect matrix
cat("Regression results:\n")
print(coef_test)

treatment_coef <- coef_test[2, 1]

p_value <- coef_test[2, 4]

se_clustered <- coef_test[2, 2]

## Compute the 95% Confidence internal for the treatment coefficient
conf_int95 <- treatment_coef + c(-1.96, 1.96)*se_clustered

## Compute variance of Y by treatment group
var_df <- data %>%
 group_by(!!sym(treatment_variable)) %>%
 summarise(variance = var(!!sym(y), na.rm = TRUE), .groups = "drop")

head(var_df)

## Extract sample sizes
n_control <- sum(data[[treatment_variable]]=="Control", na.rm = TRUE)
n_treatment <- sum(data[[treatment_variable]]=="Treatment", na.rm = TRUE)

## Extract variances
var_control <- var_df %>%
 filter(var_df[[treatment_variable]]=="Control") %>%
```

```
 pull(variance)

var_treatment <- var_df %>%
 filter(var_df[[treatment_variable]]=="Treatment") %>%
 pull(variance)

## Compute pooled SD
pooled_sd <- sqrt(((n_control - 1)* var_control + (n_treatment-1)* var_treatment) / (n_control + n_treatment - 2))

## Compute hedges G
hedges_g <- treatment_coef / pooled_sd

## Compute 95% CI for hedges G
hedges_g_ci <- conf_int95 / pooled_sd

## Print results
cat("\n### Results for outcome: ", y, "\n")
cat(" - Unadjusted difference in means: ", unadjusted_diff, "\n")
cat(" - N treatment: ", n_treatment, "\n")
cat(" - N control: ", n_control, "\n")
cat(" - Treatment coefficient: ", treatment_coef, "\n")
cat(" - p-value of treatment coefficient: ", p_value, "\n")
cat(" - 95% CI for treatment coefficient: ", conf_int95, "\n")
cat(" - Variance of ", y, "by treatment group:\n")
cat("    - Treatment: ", var_treatment, "\n")
cat("    - Control: ", var_control, "\n")
cat(" - Pooled SD: ", pooled_sd, "\n")
cat(" - Hedges G: ", hedges_g, "\n")
cat(" - 95% CI for the Hedges G: [", hedges_g_ci[1], ",", hedges_g_ci[2],"]", "\n")
```

```
 ## Return results as a list
 return(list(
  unadjusted_diff = unadjusted_diff,
  treatment_coef = treatment_coef,
  p_value = p_value,
  conf_int95 = conf_int95,
  pooled_sd = pooled_sd,
  hedges_g = hedges_g,
  hedges_g_ci = hedges_g_ci,
  variance = var_df
 ))

}



## ----CACE_results----------------------------------------------------------------------------------------------------------------------

 ### Get only observations in the analysis
  ### this is necessary because some statistics have to be computed from the analysis sample
primary_df <- master %>%
 filter(In_Primary_analysis == 1)

master %>% count(In_Primary_analysis, fsm_admin, .drop = FALSE )

table(master$compliance_20sessions)
master %>%count(compliance_20sessions, treatment_allocation)
primary_df %>%count(compliance_20sessions, treatment_allocation)
```

```
# check if these observations have NA for the outcome variable

master %>%count(compliance_20sessions, treatment_allocation, Overall_reading_score_el) # they don't.


 ### Compute results for the main CACE analysis.

CACE_results_20sessions <- CACE_model(

 data = primary_df, # dataframe to be used

 y = "Overall_reading_score_el", # outcome

 treatment_variable = "treatment_allocation",

 treatment_binary = "treatment_binary",  # treatment in binary format for regression

 compliance_variable = "compliance_20sessions",  # compliance variable

 # List of covariates:

 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin_num", "YearGroup",
"month_of_test_factor"),

 cluster = "School_ID" # cluster variable

)


# LB QA check: manual approach with ivreg

ivreg_20sessions_main <- ivreg(Overall_reading_score_el ~ compliance_20sessions + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin_num + YearGroup + month_of_test_factor

                  | treatment_binary + randomisation_batch + Overall_reading_score + eal_admin + gender_admin_num +
YearGroup + month_of_test_factor,

                  data = primary_df)


summary(ivreg_20sessions_main, vcov = vcovCL(ivreg_20sessions_main, cluster = primary_df$School_ID))

# LB QA check: we get exactly the same - OK.


# LB QA check: compute F-stat manually taking clustering into account

first_stage_20sessions_main <- lm(compliance_20sessions ~ treatment_binary + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin_num + YearGroup + month_of_test_factor, data = primary_df)

# NT's way:

summary_first_stage_20sessions_main <- summary(first_stage_20sessions_main)
```

```
f_stat_NT <- summary_first_stage_20sessions_main$fstatistic

print(f_stat_NT) # 826

# LB way to take clustering into account:

vcov_clustered <- vcovCL(first_stage_20sessions_main, cluster = primary_df$School_ID)

f_stat_LB <- waldtest(first_stage_20sessions_main, vcov = vcov_clustered, test = "F") # I get F = 151.36  (smaller, which makes sense given taking clustering into account).

f_statistic <- f_stat_LB[, "F"]

print(f_statistic)



## ----CACE_18----------------------------------------------------------------------------------------------------------------


 # Define compliance at 18 sessions

primary_df <- primary_df %>%

 mutate(

  compliance_18sessions = if_else(

   sessions_before_test >=18, 1, 0

  )

 )


primary_df %>% count(compliance_20sessions, compliance_18sessions)


# LB QA check:

primary_df %>% count(treatment_allocation, compliance_18sessions)

primary_df %>% count(treatment_allocation, sessions_before_test)


 # Call the function:

 ### Compute results for the main CACE analysis. Compliance = 18 sessions

CACE_results_18sessions <- CACE_model(

 data = primary_df, # dataframe to be used
```

```
 y = "Overall_reading_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
 compliance_variable = "compliance_18sessions",  # CHANGE compliance variable
 # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin_num", "YearGroup",
"month_of_test_factor"),
 cluster = "School_ID" # cluster variable
)

# LB QA check: manual approach with ivreg
ivreg_18sessions_main <- ivreg(Overall_reading_score_el ~ compliance_18sessions + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin_num + YearGroup + month_of_test_factor
                    | treatment_binary + randomisation_batch + Overall_reading_score + eal_admin + gender_admin_num +
YearGroup + month_of_test_factor,
                    data = primary_df)

summary(ivreg_18sessions_main, vcov = vcovCL(ivreg_18sessions_main, cluster = primary_df$School_ID))

## ----CACE_15----------------------------------------------------------------------------------------------------
 # Define compliance at 15 sessions
primary_df <- primary_df %>%
 mutate(
  compliance_15sessions = if_else(
   sessions_before_test >=15, 1, 0
  )
 )

primary_df %>% count(compliance_20sessions, compliance_15sessions)
```

```
primary_df %>% count(treatment_allocation, compliance_15sessions)


 # Call the function:
 ### Compute results for the main CACE analysis. Compliance = 18 sessions
CACE_results_15sessions <- CACE_model(
 data = primary_df, # dataframe to be used
 y = "Overall_reading_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
 compliance_variable = "compliance_15sessions",  # CHANGE compliance variable
 # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin_num", "YearGroup",
"month_of_test_factor"),
 cluster = "School_ID" # cluster variable
)


# LB QA check: manual approach with ivreg
ivreg_15sessions_main <- ivreg(Overall_reading_score_el ~ compliance_15sessions + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin_num + YearGroup + month_of_test_factor
                    | treatment_binary + randomisation_batch + Overall_reading_score + eal_admin + gender_admin_num +
YearGroup + month_of_test_factor,
                    data = primary_df)


summary(ivreg_15sessions_main, vcov = vcovCL(ivreg_15sessions_main, cluster = primary_df$School_ID))
# LB QA check: we get exactly the same - OK.


## ----CACE_20_in12----------------------------------------------------------------------------------------------------
```

```
primary_df %>% count(compliance_20sessions, compliance_20sessions_12w)


# LB QA check:

table(primary_df$compliance_20sessions_12w)

primary_df %>% count(treatment_binary, compliance_20sessions_12w) # OK.


 # Call the function:

 ### Compute results for the main CACE analysis. Compliance = 18 sessions

CACE_results_20sessions_12weeks <- CACE_model(

 data = primary_df, # dataframe to be used

 y = "Overall_reading_score_el", # outcome

 treatment_variable = "treatment_allocation",

 treatment_binary = "treatment_binary",  # treatment in binary format for regression

 compliance_variable = "compliance_20sessions_12w",  # CHANGE compliance variable

 # List of covariates:

 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin_num", "YearGroup",
"month_of_test_factor"),

 cluster = "School_ID" # cluster variable

)


# LB QA check: manual approach with ivreg

ivreg_20sessions_12w_main <- ivreg(Overall_reading_score_el ~ compliance_20sessions_12w + randomisation_batch +
Overall_reading_score + eal_admin + gender_admin_num + YearGroup + month_of_test_factor

                  | treatment_binary + randomisation_batch + Overall_reading_score + eal_admin + gender_admin_num +
YearGroup + month_of_test_factor,

                  data = primary_df)


summary(ivreg_20sessions_12w_main, vcov = vcovCL(ivreg_20sessions_12w_main, cluster = primary_df$School_ID))
```

```
## ----session_endogeneity----------------------------------------------------------------------------------------------

 ### Use only treatment observations
treatment_df <- master %>%
 filter(treatment_allocation== "Treatment")

 ### Regress the number of sessions on pupil individual characteristics: gender, FSM status, EAL and baseline NGRT

 ## Print the number of observations in the regression
 treatment_df <- treatment_df %>%
  mutate(
   in_dosage_analysis = if_else(
   !is.na(sessions_before_test) & !is.na(gender_admin_num) &
    !is.na(eal_admin) & !is.na(fsm_admin) & !is.na(Overall_reading_score), 1, 0)
  )

 session_endogeneity <- lm(
  sessions_before_test ~ gender_admin_num + eal_admin + fsm_admin + Overall_reading_score,
         data = treatment_df) # LB QA check: why not include year group here?
 cat("\n### *** Results of regressing sessions on gender_admin_num + eal_admin + fsm_admin + Overall_reading_score :
***\n")
 summary(session_endogeneity)

  session_endogeneity_check <- lm(
  sessions_before_test ~ gender_admin_num + eal_admin + fsm_admin + Overall_reading_score + YearGroup,
         data = treatment_df) # LB QA check when including it

  cat("\n### *** Results of regressing sessions on gender_admin_num + eal_admin + fsm_admin + Overall_reading_score +
YearGroup: ***\n")
```

```
  summary(session_endogeneity_check) # year group it's not a significant predictor in any case.


table(master$sessions_before_test)
master %>% count(treatment_allocation, sessions_before_test) Î


# LB QA code
 master_treatment <- master %>%
 filter(treatment_allocation == "Treatment")


session_endogeneity_qa <- lm(
  sessions_before_test ~ gender_admin_num + eal_admin + fsm_admin + Overall_reading_score,
          data = master_treatment)
summary(session_endogeneity_qa)
model_se_qa <- coeftest(session_endogeneity_qa,
                    vcov = vcovCL,
                    cluster = master_treatment$School_ID,
                    data = master_treatment)
print(model_se_qa)



 ## Compute Clustered SE by school ID using sandwich package
  # Clustered SE
 model_se <- coeftest(session_endogeneity,
                    vcov = vcovCL,
                    cluster = treatment_df$School_ID,
                    data = treatment_df)


 ## Print results of first stage
 cat("*** Regression results: *** \n")
```

```
 print(model_se)


 cat("*** Underlying counts of regression: *** \n")
 treatment_df %>% count(treatment_df$in_dosage_analysis)


cat("\n *** Quasi-poisson model results: *** \n")
session_endogeneity_bs <-glm(
   sessions_before_test ~ gender_admin_num + eal_admin + fsm_admin + Overall_reading_score,
           data = treatment_df, family = quasipoisson)
summary(session_endogeneity_bs)
 model_bs_se <- coeftest(session_endogeneity_bs,
                     vcov = vcovCL,
                     cluster = treatment_df$School_ID,
                     data = treatment_df)



## ----optimal_dosage----------------------------------------------------------------------------------------------------


  ## Filter only the observations that will be part of the analysis
   # At least 1 session -
onesession_df <- master %>%
 filter(sessions_before_test >=1 & !is.na(sessions_before_test))



  ## Start by generating the battery of percentile indicators
   # Quartile indicators
quartile_thresholds <- quantile(onesession_df$sessions_before_test,
                        probs = c(#DELETED FOR CLEARANCE
```

```
                    ),
                    na.rm = TRUE)
print(quartile_thresholds)

  #  Quintile indicators
quintile_thresholds <- quantile(onesession_df$sessions_before_test,
                    probs = c(0.2, 0.4, 0.6, 0.8),
                    na.rm = TRUE)
print(quintile_thresholds)

  # Create indicators
treatment_df <- treatment_df %>%

 mutate(
  quartile_1 = if_else(sessions_before_test <= quartile_thresholds[1] & sessions_before_test >=1, 1, 0),
  quartile_2 = if_else(sessions_before_test > quartile_thresholds[1] &
                sessions_before_test <= quartile_thresholds[2], 1, 0),
  quartile_3 = if_else(sessions_before_test > quartile_thresholds[2] &
                sessions_before_test <= quartile_thresholds[3], 1, 0),
  quartile_4 = if_else(sessions_before_test > quartile_thresholds[3], 1, 0)
 ) %>%
   # Quintiles:
  mutate(
  quintile_1 = if_else(sessions_before_test <= quintile_thresholds[1] & sessions_before_test >=1, 1, 0),
  quintile_2 = if_else(sessions_before_test > quintile_thresholds[1] &
                sessions_before_test <= quintile_thresholds[2], 1, 0),
  quintile_3 = if_else(sessions_before_test > quintile_thresholds[2] &
                sessions_before_test <= quintile_thresholds[3], 1, 0),
  quintile_4 = if_else(sessions_before_test > quintile_thresholds[3] &
```

```
                sessions_before_test <= quintile_thresholds[4], 1, 0),
  quintile_5 = if_else(sessions_before_test > quintile_thresholds[4], 1, 0)
 ) %>%
 mutate(
  zero_sessions = if_else(sessions_before_test == 0, 1, 0)
 )

sum(is.na(treatment_df$sessions_before_test))
sum(is.na(treatment_df$quartile_1))
sum(is.na(treatment_df$zero_sessions))

sum(treatment_df$zero_sessions)
treatment_df %>% count(zero_sessions)
treatment_df %>% count(sessions_before_test)
treatment_df %>% count(sessions_before_test, quartile_1)

treatment_df %>% count(sessions_before_test, zero_sessions)

# Check distribution of indicators
treatment_df %>%
 summarise(across(starts_with(c("quartile_", "quintile_", "zero_sessions")), ~sum(., na.rm = TRUE))) %>%
 glimpse

  # Create a variable equal to the increase in NGRT
treatment_df <- treatment_df %>% # NTB> changed the data
 mutate(NGRT_diff = Overall_reading_score_el - Overall_reading_score) %>%
 mutate(NGRT_diff = if_else(is.na(Overall_reading_score_el) | is.na(Overall_reading_score), NA, NGRT_diff)
     )
```

```
summary(treatment_df$NGRT_diff)


  # Inspect visually by graphing a scatter plot
treatment_df_2 <- treatment_df %>%
 filter(sessions_before_test >=1 & !is.na(sessions_before_test))


ggplot(treatment_df_2, aes(x = 1:length(NGRT_diff), y = NGRT_diff)) +
 geom_point(color = "blue", size= 3, alpha = 0.7) +
 labs(
  x = "Index",
  y = "NGRT_diff") +
 theme_minimal()


treatment_df %>% count(quintile_5, sessions_before_test)



## ----quartiles----------------------------------------------------------------------------------------------------------------


 # Quartile regression
quartile_model <- lm(NGRT_diff ~ quartile_1 + quartile_2 + quartile_3 + quartile_4 +
                Overall_reading_score + eal_admin + gender_admin_num + fsm_admin,
               data = treatment_df)


 # Get SE
quartile_robust  <- coeftest(quartile_model,
                   vcov = vcovHC(quartile_model, type = "HC1"))


print(quartile_robust)
```

```
 # Create a tidy dataframe with results
quartile_results <- data.frame(
 term = rownames(quartile_robust),
 estimate = quartile_robust[,1],
 std.error = quartile_robust[,2],
 statistic = quartile_robust[,3],
 p.value = quartile_robust[,4]
) %>%
 mutate(
  conf.low = estimate - 1.96*std.error,
  conf.high = estimate + 1.96*std.error
 )

 # Plot the result against



## ----quintiles----------------------------------------------------------------------------------------------------------------

 # Quintile regression
quintile_model <- lm(NGRT_diff ~ quintile_1 + quintile_2 + quintile_3 + quintile_4 + quintile_5 + Overall_reading_score +
eal_admin + gender_admin_num + fsm_admin,
                data = treatment_df)


treatment_df %>% count(sessions_before_test, quintile_1)
treatment_df %>%  count(zero_sessions, quintile_1, sessions_before_test)
treatment_df %>% count(zero_sessions, quintile_1)


 # Get SE
quintile_robust  <- coeftest(quintile_model,
                     vcov = vcovHC(quintile_model, type = "HC1"))
```

```
print(quintile_robust)

 # Create a tidy dataframe with results
quintile_results <- data.frame(
 term = rownames(quintile_robust),
 estimate = quintile_robust[,1],
 std.error = quintile_robust[,2],
 statistic = quintile_robust[,3],
 p.value = quintile_robust[,4]
) %>%
 mutate(
  conf.low = estimate - 1.96*std.error,
  conf.high = estimate + 1.96*std.error
 )

  # Convert to ordered factors
quintile_results$quintile_num <- as.numeric(gsub("quintile_", "", quintile_results$term))
quintile_results$x_label <- paste0(quintile_results$quintile_num*20, "%")

intercept <- quintile_results[1, "estimate"]

quintile_results_filtered2 <- quintile_results %>%
 filter(grepl("quintile_[1-5]", term) | grepl("Intercept", term))

quintile_results_filtered2 <- quintile_results_filtered2 %>%
 mutate(
  predicted_diff = if_else(!is.na(quintile_num),intercept + estimate, estimate),
  predicted_ci_low = if_else(!is.na(quintile_num), intercept + conf.low, conf.low),
  predicted_ci_high = if_else(!is.na(quintile_num), intercept + conf.high, conf.high)
```

```
 )


quintile_results_filtered2 <- quintile_results_filtered2 %>%
 mutate(
  predicted_diff = if_else(is.na(quintile_num), estimate, predicted_diff),
  quintile_num = if_else(term == "(Intercept)", 0, quintile_num),
  x_label = if_else(term == "(Intercept)", "0", x_label)
 )



 ## Need to fix label
str(quintile_results_filtered2$quintile_num)
str(quintile_results_filtered2$x_label)
quintile_results_filtered2$quintile_num <- as.factor(quintile_results_filtered2$quintile_num)

## Generate another label
quintile_results_filtered2 <- quintile_results_filtered2 %>%
 mutate(
  x_label_2 = case_when(
   quintile_num == 0 ~ "0",
   quintile_num == 1 ~ "1-20",
   quintile_num == 2 ~ "21-24",
   quintile_num == 3 ~ "25-27",
   quintile_num == 4 ~ "28-31",
   quintile_num == 5 ~ "32-51",
   .default = NA
  )
 )
```

```
graph2 <- ggplot(quintile_results_filtered2, aes(x = quintile_num, y = predicted_diff, group = 1)) +
  geom_line() +
  geom_point(size = 3) +
  geom_errorbar(aes(ymin = predicted_ci_low, ymax = predicted_ci_high), width = 0.2) +
  labs(title = "Increase in NGRT overall reading score, by dosage",
       x = "Quintile of number of RR sessions attended",
       y = "Change in NGRT score") +
  scale_x_discrete(labels = quintile_results_filtered2$x_label) +
  theme_minimal()
print(graph2)

graph3 <- ggplot(quintile_results_filtered2, aes(x = quintile_num, y = predicted_diff, group = 1)) +
  geom_line() +
  geom_point(size = 3) +
  geom_errorbar(aes(ymin = predicted_ci_low, ymax = predicted_ci_high), width = 0.2) +
  labs(title = "Increase in NGRT overall reading score, by dosage",
       x = "Number of RR sessions attended",
       y = "Change in NGRT score") +
  scale_x_discrete(labels = quintile_results_filtered2$x_label_2) +
  theme_minimal()
print(graph3)
```

Appendix Q: Missing data analysis code

```
## ----setup2, echo = FALSE, results='hide'----------------------------------------------------------------------------

## Load packages
library(tidyverse)
library(glm2) # for all regression models if needed
library(broom)
library(sandwich) # for clustered standard errors
library(zoo)
library(lmtest)
library(rmarkdown)
library(AER) # For IV regressions
library(knitr) # TO display interactive dataframes when rendering
library(Hmisc) # For the function that computes Lee bounds

## Clean current working environment
rm(list = ls())

## Check current directory - It should be the same folder of the quarto document
getwd()



## ----load_data, echo = FALSE, results='hide'-------------------------------------------------------------------------
## Load main dataset
#DELETED FOR CLEARANCE

master <- master %>%
  select(-...1)
```

```
## ----covariates, max.print=50, width=50-----------------------------------------------------------------------------------

  ###  Analysis on NGRT overall score ###
 overall_reading_covs <- master %>%
 filter(!is.na(Overall_reading_score_el)) %>%
   group_by(treatment_allocation) %>%
   summarise(
    variable = "Overall_reading_score_el",
    total_obs = n(),
    missing_fsm = sum(is.na(fsm_admin)),
    missing_percent_fsm = round(missing_fsm / total_obs*100, 2),
    missing_gender = sum(is.na(gender_admin)),
    missing_percent_gender = round(missing_gender / total_obs*100, 2),
    missing_eal = sum(is.na(eal_admin)),
    missing_percent_eal = round(missing_eal / total_obs*100, 2),
    missing_year = sum(is.na(YearGroup)),
    missing_percent_year = round(missing_year / total_obs*100, 2),
    missing_baseline = sum(is.na(Overall_reading_score)),
    missing_percent_baseline = round(missing_baseline / total_obs*100, 2)
   )
 cat("MIssing covariates in primary analysis (NGRT overall score)")
 kable(overall_reading_covs)


## ----manual---------------------------------------------------------------------------------------------------------------
```

```
sum(is.na(master$fsm_admin) & !is.na(master$Overall_reading_score_el))

sum(is.na(master$gender_admin) & !is.na(master$Overall_reading_score_el))

sum(is.na(master$eal_admin) & !is.na(master$Overall_reading_score_el))



## ----covariates_2----------------------------------------------------------------------------------------------------
  ### Analysis on NGRT PC score ###
 PC_covs <- master %>%
 filter(!is.na(PC_score_el)) %>%
   group_by(treatment_allocation) %>%
   summarise(
    variable = "PC_score_el",
    total_obs = n(),
    missing_fsm = sum(is.na(fsm_admin)),
    missing_percent_fsm = round(missing_fsm / total_obs*100, 2),
    missing_gender = sum(is.na(gender_admin)),
    missing_percent_gender = round(missing_gender / total_obs*100, 2),
    missing_eal = sum(is.na(eal_admin)),
    missing_percent_eal = round(missing_eal / total_obs*100, 2),
    missing_year = sum(is.na(YearGroup)),
    missing_percent_year = round(missing_year / total_obs*100, 2),
    missing_baseline = sum(is.na(PC_score)),
    missing_percent_baseline = round(missing_baseline / total_obs*100, 2)
   )
 cat("MIssing covariates in secondary analysis (NGRT PC score)")
 kable(PC_covs)
```

```
  ### Analysis on NGRT SC score ###
SC_covs <- master %>%
filter(!is.na(SC_score_el)) %>%
  group_by(treatment_allocation) %>%
  summarise(
    variable = "SC_score_el",
    total_obs = n(),
    missing_fsm = sum(is.na(fsm_admin)),
    missing_percent_fsm = round(missing_fsm / total_obs*100, 2),
    missing_gender = sum(is.na(gender_admin)),
    missing_percent_gender = round(missing_gender / total_obs*100, 2),
    missing_eal = sum(is.na(eal_admin)),
    missing_percent_eal = round(missing_eal / total_obs*100, 2),
    missing_year = sum(is.na(YearGroup)),
    missing_percent_year = round(missing_year / total_obs*100, 2),
    missing_baseline = sum(is.na(SC_score)),
    missing_percent_baseline = round(missing_baseline / total_obs*100, 2)
  )
cat("MIssing covariates in secondary analysis (NGRT SC score)")
kable(SC_covs)


    ### Analysis on KS2 score ###
KS2_covs <- master %>%
filter(!is.na(KS2_READSCORE_num)) %>%
  group_by(treatment_allocation) %>%
  summarise(
    variable = "KS2_READSCORE_num",
    total_obs = n(),
    missing_fsm = sum(is.na(fsm_admin)),
```

```
    missing_percent_fsm = round(missing_fsm / total_obs*100, 2),
    missing_gender = sum(is.na(gender_admin)),
    missing_percent_gender = round(missing_gender / total_obs*100, 2),
    missing_eal = sum(is.na(eal_admin)),
    missing_percent_eal = round(missing_eal / total_obs*100, 2),
    missing_year = sum(is.na(YearGroup)),
    missing_percent_year = round(missing_year / total_obs*100, 2),
    missing_baseline = sum(is.na(Overall_reading_score)),
    missing_percent_baseline = round(missing_baseline / total_obs*100, 2)
  )
cat("MIssing covariates in secondary analysis (KS2 reading score)")
kable(KS2_covs)



## ----missing_outcome----------------------------------------------------------------------------------------------------------

 # Randomised observations missing outcome data by trial arm and in total #
 ### Calculate missing observations for each outcome by treatment arm and overall

  ###  Analysis on NGRT overall score ###
  overall_reading_byarm <- master %>%
   group_by(treatment_allocation) %>%
   summarise(
    variable = "Overall_reading_score_el",
    missing_obs = sum(is.na(Overall_reading_score_el)),
    total_obs = n(),
    missing_percent = round(missing_obs / total_obs*100, 2),
   )
```

```
overall_reading_total <- master %>%
 summarise(
  variable = "Overall_reading_score_el",
  missing_obs = sum(is.na(Overall_reading_score_el)),
  total_obs = n(),
  missing_percent = round(missing_obs / total_obs*100, 2),
) %>%
 mutate(
   treatment_allocation  = "Total") %>%
 relocate(treatment_allocation)

 ###  Analysis on NGRT PC score ###
PC_byarm <- master %>%
 group_by(treatment_allocation) %>%
 summarise(
  variable = "PC_score_el",
  missing_obs = sum(is.na(PC_score_el)),
  total_obs = n(),
  missing_percent = round(missing_obs / total_obs*100, 2),
)

PC_total <- master %>%
 summarise(
  variable = "PC_score_el",
  missing_obs = sum(is.na(PC_score_el)),
  total_obs = n(),
  missing_percent = round(missing_obs / total_obs*100, 2),
) %>%
```

```
 mutate(
  treatment_allocation  = "Total") %>%
 relocate(treatment_allocation)

   ###  Analysis on NGRT SC score ###
SC_byarm <- master %>%
 group_by(treatment_allocation) %>%
 summarise(
  variable = "SC_score_el",
  missing_obs = sum(is.na(SC_score_el)),
  total_obs = n(),
  missing_percent = round(missing_obs / total_obs*100, 2),
)

SC_total <- master %>%
 summarise(
  variable = "SC_score_el",
  missing_obs = sum(is.na(SC_score_el)),
  total_obs = n(),
  missing_percent = round(missing_obs / total_obs*100, 2),
) %>%
 mutate(
  treatment_allocation  = "Total") %>%
 relocate(treatment_allocation)

###  Analysis on KS2 reading score ###
KS2_byarm <- master %>%
 filter(YearGroup == 6) %>%
 group_by(treatment_allocation) %>%
```

```
  summarise(
   variable = "KS2_READSCORE_num",
   missing_obs = sum(is.na(KS2_READSCORE_num)),
   total_obs = n(),
   missing_percent = round(missing_obs / total_obs*100, 2),
  )

 KS2_total <- master %>%
 filter(YearGroup == 6) %>%
  summarise(
   variable = "KS2_READSCORE_num",
   missing_obs = sum(is.na(KS2_READSCORE_num)),
   total_obs = n(),
   missing_percent = round(missing_obs / total_obs*100, 2),
 ) %>%
  mutate(
    treatment_allocation  = "Total") %>%
  relocate(treatment_allocation)


 # Merge results
 missing_outcome <-bind_rows(overall_reading_byarm, overall_reading_total, PC_byarm, PC_total, SC_byarm, SC_total,
KS2_byarm, KS2_total)


kable(missing_outcome)




## ----logit----------------------------------------------------------------------------------------------------------
 # Create missing indicator pf primary outcome
master <- master %>%
```

```
 mutate(
  m_overall = if_else(is.na(Overall_reading_score_el), 1, 0),
  treatment_binary = if_else(treatment_allocation=="Treatment", 1, 0)
 )
master %>% count(m_overall)

 # Regress missingness on covariates and treatment assingment
model_missingness <- glm(m_overall ~ treatment_binary + Overall_reading_score + fsm_admin + eal_admin + gender_admin,
            data = master,
           family = binomial(link="logit") # Use logit
            )

summary(model_missingness)



## ----analysis_function-----------------------------------------------------------------------------------------------------
 ## Set up the function:



 ### Compute total observations per treatment arm (will be needed later)
N_treatment <- sum(master$treatment_allocation=="Treatment", na.rm = TRUE)
N_control <- sum(master$treatment_allocation=="Control", na.rm = TRUE)

 ## Set up the function
treatment_effect_ols <- function(data, y, treatment_variable, treatment_binary, covariates = NULL, cluster) {

 ## Set regression formula
 regression_formula <- as.formula(
  paste(y, "~", treatment_binary,
```

```
    if (!is.null(covariates)) paste("+", paste(covariates, collapse = " + ")) else "")
)

print(regression_formula)

## Run OLS regression - use glm2 package
model <- glm(regression_formula,
        data = data,
        family = gaussian # Use linear regression - OLS
        )

## Compute Clustered SE by school ID using sandwich package
clustered_se <- vcovCL(model, cluster = data[[cluster]], type = "HC1")

## Extract treatment coefficients and p-value from matrix
coef_test <- coeftest(model, vcov = clustered_se)
print(rownames(coef_test))

  ## Inspect matrix
print("Regression results:")
print(coef_test)

treatment_coef <- coef_test[2, 1]

p_value <- coef_test[2, 4]

se_clustered <- coef_test[2, 2]

## Compute the 95% Confidence internal for the treatment coefficient
```

```
 conf_int95 <- treatment_coef + c(-1.96, 1.96)*se_clustered


 ## Compute the raw means, CI of the means and unadjusted difference
stats_df <- data %>%
  group_by(!!sym(treatment_variable)) %>%
   # Ignore missing values (we'll only use observations with non-missing values, as otherwise they are dropped from the regression)
 summarise(
  mean_y = mean(!!sym(y), na.rm = TRUE),
  sd_y = sd(!!sym(y), na.rm = TRUE),
  n = n(),
  se = sd_y / sqrt(n),
  ci_lower = mean_y - 1.96 * se,
  ci_upper = mean_y + 1.96 * se,
  .groups = "drop")

  # Extract raw means and  CI
 mean_control <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Control") %>%
  pull(mean_y)

 mean_treatment <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Treatment") %>%
  pull(mean_y)

  unadjusted_diff <- mean_treatment - mean_control

 ci_lower_control <- stats_df %>%
  filter(stats_df[[treatment_variable]]=="Control") %>%
  pull(ci_lower)
```

```
ci_lower_treatment <- stats_df %>%
 filter(stats_df[[treatment_variable]]=="Treatment") %>%
 pull(ci_lower)

ci_upper_control <- stats_df %>%
 filter(stats_df[[treatment_variable]]=="Control") %>%
 pull(ci_upper)

ci_upper_treatment <- stats_df %>%
 filter(stats_df[[treatment_variable]]=="Treatment") %>%
 pull(ci_upper)

 ## Extract sample sizes
n_control <- sum(data[[treatment_variable]]=="Control", na.rm = TRUE)
n_treatment <- sum(data[[treatment_variable]]=="Treatment", na.rm = TRUE)
total_n <- n_control + n_treatment

## Extract missing sizes
n_control_missing <- N_control - n_control
n_treatment_missing <- N_treatment - n_treatment

## Compute variance of Y by treatment group
var_df <- data %>%
 group_by(!!sym(treatment_variable)) %>%
 summarise(variance = var(!!sym(y), na.rm = TRUE), .groups = "drop")

head(var_df)
```

```
## Extract variances
var_control <- var_df %>%
 filter(var_df[[treatment_variable]]=="Control") %>%
 pull(variance)

var_treatment <- var_df %>%
 filter(var_df[[treatment_variable]]=="Treatment") %>%
 pull(variance)

## Compute pooled SD
pooled_sd <- sqrt(((n_control - 1)* var_control + (n_treatment-1)* var_treatment) / (n_control + n_treatment - 2))

pooled_var <- pooled_sd^2

## Compute hedges G
hedges_g <- treatment_coef / pooled_sd

## Compute 95% CI for hedges G
hedges_g_ci <- conf_int95 / pooled_sd

## Print results
cat("\n### Results for outcome: ", y, "\n")
cat(" - Unadjusted difference in means: ", unadjusted_diff, "\n")
cat(" - N treatment: ", n_treatment, "\n")
cat(" - N treatment (missing): ", n_treatment_missing, "\n")
cat(" - N control: ", n_control, "\n")
cat(" - N control (missing): ", n_control_missing, "\n")
cat(" - Total N: ", total_n, "\n")
cat(" - Mean treatment: ", mean_treatment, "\n")
```

```
cat(" - Mean control: ", mean_control, "\n")
cat(" - 95% CI lower bound treatment: ", ci_lower_treatment, "\n")
cat(" - 95% CI upper bound treatment: ", ci_upper_treatment, "\n")
cat(" - 95% CI lower bound control: ", ci_lower_control, "\n")
cat(" - 95% CI upper bound control: ", ci_upper_control, "\n")

cat(" - Treatment coefficient: ", treatment_coef, "\n")
cat(" - p-value of treatment coefficient: ", p_value, "\n")
cat(" - 95% CI for treatment coefficient: ", conf_int95, "\n")
cat(" - Variance of ", y, "by treatment group:\n")
cat("    - Treatment: ", var_treatment, "\n")
cat("    - Control: ", var_control, "\n")
cat(" - Pooled SD: ", pooled_sd, "\n")
cat(" - Pooled variance: ", pooled_var, "\n")
cat(" - Hedges G: ", hedges_g, "\n")
cat(" - 95% CI for the Hedges G: [", hedges_g_ci[1], ",", hedges_g_ci[2],"]", "\n")

## Return results as a list
return(list(
  unadjusted_diff = unadjusted_diff,
  treatment_coef = treatment_coef,
  p_value = p_value,
  conf_int95 = conf_int95,
  pooled_sd = pooled_sd,
  hedges_g = hedges_g,
  hedges_g_ci = hedges_g_ci,
  variance = var_df
))
```

```
}



## ----sens_analysis-----------------------------------------------------------------------------------------------------------


 ### Get only observations in the analysis

  ### this is necessary because some statistics have to be computed from the analysis sample
master <- master %>%
 mutate(
  in_sens_analysis = if_else(
   !is.na(treatment_allocation) & !is.na(randomisation_batch) &
    !is.na(Overall_reading_score) & !is.na(Overall_reading_score_el) & !is.na(YearGroup) & !is.na(gender_admin) &
    !is.na(eal_admin) & !is.na(month_of_test) & !is.na(fsm_admin), 1, 0))

sens_df <- master %>%
 filter(in_sens_analysis == 1)


 ### Compute results of main model adding fsm status


  ## Call the function - SUbstitute for the right covariates in the model
results_sens_analysis <- treatment_effect_ols(
 data = sens_df, # dataframe to be used
 y = "Overall_reading_score_el", # outcome
 treatment_variable = "treatment_allocation",
 treatment_binary = "treatment_binary",  # treatment in binary format for regression
  # List of covariates:
 covariates = c("randomisation_batch", "Overall_reading_score", "eal_admin", "gender_admin", "YearGroup", "month_of_test",
"fsm_admin"),
 cluster = "School_ID" # cluster variable
)
```

```
## ----lee_bounds_function--------------------------------------------------------------------------------------------

## Prepare the data to be used
 # treat = binary treatment
 # selection variable = 1 if outcome is observed
 # outcome = outcome * selection
leedata <- master %>%
 rename(
  treat = treatment_binary,
  outcome = Overall_reading_score_el
 ) %>%
 mutate(
  selection = if_else(m_overall == 0, 1, 0)
 ) %>%
 select(Pupil_ID, treat, outcome, selection)

## Code the function
basic_lee_bound<-function(leedata, # data to be used - it has to be in the format specified above
                treat_helps=NULL, # NULL = No assumotion about treatment effect on sample selection
                # TRUE = Assujmes treatment increases selection probability
                # FALSE = assumes treatment decreases selection probability
                ...) {
 print(treat_helps)
 d<-leedata$treat
 s<-leedata$selection
 sy<-leedata$outcome
```

```
if ("weights" %in% colnames(leedata)) {
 weights<-leedata$weights
} else {
 weights<-rep(1,length(d))
}

# s: binary (1/0)selection
# sy: outcome; sy=0 if selection s=0, sy=y otherwise
# normalize weights; assume weights are positive
treat_size<-sum(d==1)
control_size<-sum(d==0)
prop_control_nonmissing<-weighted.mean(s[d==0]==1,weights[d==0])
prop_treat_nonmissing<-weighted.mean(s[d==1]==1,weights[d==1])

p0<-prop_control_nonmissing/prop_treat_nonmissing
#print(p0)
if (!is.null(treat_helps)) {
 if (treat_helps==TRUE) {
  p0<-sapply(p0,min,1)
 }
 if (treat_helps==FALSE) {
  p0<-sapply(p0,max,1)
 }

}
if (prop_treat_nonmissing==0|is.na(p0)) {
 leebounds_result=list(lower_bound=NA,upper_bound=NA)
 return(leebounds_result)
```

```
}
if (p0<=1) {
 trim_group_inds<-(d==1)
 nontrim_group_inds<-(d==0)
 p0_star = p0
} else {
 p0_star = 1/p0
 trim_group_inds<-(d==0)
 nontrim_group_inds<-(d==1)
}

flag_NA<-FALSE
if (sum(trim_group_inds &s==1)==0) {
 # stop("No identification: no observed data in trimmed group")
 flag_NA<-TRUE
}
if (sum(nontrim_group_inds  &s==1)==0) {
 flag_NA<-TRUE
 # stop("No identification: no observed data in nontrimmed group")
}
if (flag_NA) {
 leebounds_result=list(lower_bound=NA,upper_bound=NA)
 return(leebounds_result)
} else {
 y_trim<-sy[trim_group_inds &s==1]
 y_nontrim<-sy[nontrim_group_inds & s==1]

 weights_trim<-weights[trim_group_inds &s==1]
```

```
  if ("weights" %in% colnames(leedata)) {
   yp0<-reldist::wtd.quantile(y_trim,q=p0_star,weight=weights[trim_group_inds &s==1])
   y1p0<-reldist::wtd.quantile(y_trim,q=1-p0_star,weight=weights[trim_group_inds &s==1])

  } else {
   yp0<-stats::quantile(y_trim,p0_star)
   y1p0<-stats::quantile(y_trim,1-p0_star)

  }

  trimmed_mean_upper<-weighted.mean(y_trim[y_trim>=y1p0],weights_trim[y_trim>=y1p0])
  trimmed_mean_lower<-weighted.mean(y_trim[y_trim<=yp0],weights_trim[ y_trim<=yp0])

  if (p0<=1) {
   upper_bound_effect<- trimmed_mean_upper-weighted.mean(y_nontrim,weights[ nontrim_group_inds & s==1])
   lower_bound_effect<- trimmed_mean_lower-weighted.mean(y_nontrim,weights[ nontrim_group_inds & s==1])
  } else {
   upper_bound_effect<- weighted.mean(y_nontrim,weights[ nontrim_group_inds & s==1])-trimmed_mean_lower
   lower_bound_effect<- weighted.mean(y_nontrim,weights[ nontrim_group_inds & s==1])-trimmed_mean_upper
  }

  leebounds_result<-list(lower_bound=lower_bound_effect,upper_bound=upper_bound_effect)


return(list(lower_bound=lower_bound_effect,upper_bound=upper_bound_effect,p0=prop_control_nonmissing/prop_treat_no
nmissing,
        trimmed_mean_upper=trimmed_mean_upper,trimmed_mean_lower=trimmed_mean_lower,
        mean_no_trim=mean(y_nontrim),
        odds=treat_size/control_size,
        yp0=yp0,
```

```
            y1p0=y1p0,
            s0=prop_control_nonmissing,
            s1=prop_treat_nonmissing,
            prop0=mean(d==0),
            prop1=mean(d==1)))
 }
}

basic_bounds <- basic_lee_bound(leedata,
                        treat_helps = FALSE)
cat("Basic_bounds: ", basic_bounds[[1]],basic_bounds[[2]])




## ----trimmed_bounds-----------------------------------------------------------------------------------------------------------

 ## Preliminary step: transform baseline attainment into a categorical variable
  # Break the distribution into 4 quartiles
quartile_thresholds <- quantile(master$Overall_reading_score,
                        probs = c(0.25, 0.5, 0.75),
                        na.rm = TRUE)
cat("Quartile thresholds of baseline attainment: ")
print(quartile_thresholds)

  # Create indicators
master<- master %>%
 mutate(
  baseline = case_when(
   Overall_reading_score <= quartile_thresholds[1] ~ 1,
```

```
    Overall_reading_score > quartile_thresholds[1] &  Overall_reading_score <= quartile_thresholds[2] ~ 2,
    Overall_reading_score > quartile_thresholds[2] &  Overall_reading_score <= quartile_thresholds[3] ~ 3,
    Overall_reading_score >= quartile_thresholds[3] ~ 4,
    .default = 0 # For missing observations
   )
 )
cat("Check how many pupils are in each of the 4 baseline attainment quartiles: ")

master %>% count(baseline)
master <- master %>%
 mutate(
  baseline = as.factor(baseline)
 )

  # These may be too many cells. We'll create another discrete indicator to divide pupils between those that had Overall attainment below and above the mean

base_mean <- mean(master$Overall_reading_score, na.rm = TRUE)

 master <- master %>%
 mutate(
  baseline_binary = case_when(
   Overall_reading_score <= base_mean ~ 1,
   Overall_reading_score > base_mean ~ 2,
   .default = 0
 ))
  master <- master %>%
  mutate(
   baseline_binary = as.factor(baseline_binary)
  )
```

```
 cat("Check how many pupils are in the two baseline attainment categories (above and below the mean): ")

master %>% count(baseline_binary)


 ## Create a grouping variable
leedata2 <- master %>%
 rename(
  treat = treatment_binary,
  outcome = Overall_reading_score_el
 ) %>%
 mutate(
  selection = if_else(m_overall == 0, 1, 0)
 ) %>%
 select(Pupil_ID, treat, outcome, selection, fsm_admin, baseline)

leedata3 <- leedata2 %>%
 group_by(fsm_admin, baseline) %>%
 mutate(category = paste(treat, fsm_admin, baseline, sep = "-"),
      group = cur_group_id()) %>%
 ungroup()

 cat("Count the number of pupils that are in each cell, where each cell is a combination of FSM status, and baseline quartile: ")

leedata3 %>% count(group)

 cat("Cross-tabulate with selection variable: ")

leedata3 %>% count(group, selection)
```

```
leedata3 <- leedata3 %>%
 filter(group!=11)

 ## Save another dataset with less cells
leedata4 <- master %>%
 rename(
  treat = treatment_binary,
  outcome = Overall_reading_score_el
 ) %>%
 mutate(
  selection = if_else(m_overall == 0, 1, 0)
 ) %>%
 select(Pupil_ID, treat, outcome, selection, fsm_admin, baseline_binary)

leedata4 <- leedata4 %>%
 group_by(fsm_admin, baseline_binary) %>%
 mutate(category = paste(treat, fsm_admin, baseline_binary, sep = "-"),
      group = cur_group_id()) %>%
 ungroup()

cat("Count the number of pupils that are in each cell, where each cell is a combination of FSM status, and baseline binary category: ")

 cat("Cross-tabulate with selection variable: ")

leedata4 %>% count(group, selection, useNA = "ifany")

cat("Cross-tabulate with treatment variable: ")
leedata4 %>% count(group, treat, useNA = "ifany")
```

```
leedata4 <- leedata4 %>%
 filter(group!=7)



## ----trimmed_function----------------------------------------------------------------------------------------------------
 ###Set up the function for the Lee bounds


  ## Auxiliary functions
GetBounds<-function(x) {
 return(c(x$lower_bound,x$upper_bound))
}

GetThresh<-function(x) {
 return(x$p0)
}

discrete_bound<-function(leedata, # data
                         treat_helps=NULL,
                         group) {  # grouping variable that determines the cells


  ## Add weights if they don't exist in the dataframe
 if (!("weights" %in% colnames(leedata))) {
     leedata <- leedata %>%
       mutate(
         weights = 1
       )
 }
```

```
### construct groups

leedata<-leedata[,c("treat","selection","outcome","group","weights")]


leedata<-group_by(leedata,group) %>% arrange(group)

leedata<-group_by(leedata,group)


p0s_by_group<-matrix(unlist(lapply(group_map(leedata,basic_lee_bound,treat_helps=treat_helps,
.keep=TRUE),GetThresh)),ncol=1,byrow=TRUE)


## calculate group-specific bounds and p0s

bounds_by_group<-matrix(unlist(lapply(group_map(leedata,basic_lee_bound,treat_helps=treat_helps,
.keep=TRUE),GetBounds)),ncol=2,byrow=TRUE)




  if (treat_helps==TRUE) {

    inds<-leedata$selection==1 & leedata$treat ==0


    pmf<-
wtd.table(leedata$group[inds],weights=leedata$weights[inds])/sum(wtd.table(leedata$group[inds],weights=leedata$weights[
inds]))




  } else {

    inds<-leedata$selection==1 & leedata$treat ==1

    pmf<-
wtd.table(leedata$group[inds],weights=leedata$weights[inds])/sum(wtd.table(leedata$group[inds],weights=leedata$weights[
inds]))


  }


print('pmf - Probability functions')
```

```
  print(pmf)
  print('bounds of treatment effect by group')
  print(bounds_by_group)
  print(bounds_by_group[1:nrow(bounds_by_group)-1, ])

  bounds<-pmf %*% bounds_by_group
  if (treat_helps){
   p0<-pmf%*%sapply(p0s_by_group,min,0.99)
  } else {
   p0<-pmf%*%sapply(p0s_by_group,max,1.01)
  }



  return(list(lower_bound=bounds[1],
              upper_bound=bounds[2],
              p0=p0))
}

my_wtd_table <- function(x, weights = NULL) {
  # Call the original function
  result <- Hmisc::wtd.table(x, weights = weights)

  # ENsure the result is in the expected format
  if(is.list(result) && !is.null(result$sum.of.weights)){
    return(result$sum.of.weights)
  } else {
    return(result)
  }
}
```

```
# Assign the wrapper to the global environment
assign("wtd.table", my_wtd_table, envir = .GlobalEnv)


 ## Evaluate the function
trimmed_bounds <- discrete_bound(leedata = leedata3 %>% filter(group != 11), treat_helps = FALSE, group)
cat("Trimmed bounds: ", trimmed_bounds[[1]], trimmed_bounds[[2]])

## Compare them to the basic bounds
cat("Basic_bounds: ", basic_bounds[[1]],basic_bounds[[2]])
```

Appendix R: Histograms code

```
## ----echo = FALSE----------------------------------------------------------------------------------------------------

## Load packages
library(tidyverse)
library(ggplot2)
library(rmarkdown)
library(knitr)

## Clean current working environment
rm(list = ls())

## Check current directory - It should be the same folder of the quarto document
#getwd()



## ----load_data, echo = FALSE-----------------------------------------------------------------------------------------
## Load main dataset
#DELETED FOR CLEARANCE

master <- master %>%
  select(-...1)

# Turn off the tibble printing restrictions to show all rows
options(
  tibble.print_max = Inf,
  tibble.print_min = Inf,
  tibble_width = Inf,
```

```
  dplyr.width = Inf,
  pillar.min_chars= Inf)




## ----loop----------------------------------------------------------------------------------------------------------------------------------


  ## Set up the Function
create_histogram <- function(
    data = master,
    variable,
    bin_width = 20,
    x_label = NULL,
    filename = NULL) {



  # Establish bins of 20 - SRS requires to show the underlying counts of histograms. No bin can have less than 10 observations to be cleared


  ## Extract the variable as vector
  variable_vector <- data[[variable]]


  # Compute the SD
  mean_val <- mean(variable_vector, na.rm = TRUE)
  sd_val <- sd(variable_vector, na.rm = TRUE)
  n_val <- sum(!is.na(variable_vector))


  cat("\nBasic Statistics for", variable, ":\n")
  cat("Mean:", round(mean_val, 2), "\n")
```

```
cat("Standard Deviation:", round(sd_val, 2), "\n")
cat("N:", n_val, "\n")

# Set range of data
min_y <- floor(min(variable_vector, na.rm = TRUE) / bin_width) * bin_width
max_y <- ceiling(max(variable_vector, na.rm = TRUE) / bin_width) * bin_width
print(paste("Data range:", min_y, "to", max_y))

# Create bins
bin_breaks <- seq(min_y, max_y, by= bin_width)
#print(bin_breaks)

# Tabulate data
y_tabulation <- data %>%
 mutate(bin = cut(.data[[variable]],
            breaks = bin_breaks, include.lowest = TRUE,
            labels = paste(head(bin_breaks, -1),
                     tail(bin_breaks, -1),
                     sep = "-"))) %>%
 group_by(bin) %>%
 summarise(count = n(), .groups = 'drop')
#print("Tabulation of variable values:")
#print(y_tabulation)

# Add a column to indicate if the bin should be censored
y_tabulation <- y_tabulation %>%
 mutate(censored = if_else(count<10, "Yes", "No")) %>%
 mutate(censored_count = if_else(censored == "No", count, NA))
censored_tabulation <- y_tabulation %>%
```

```
 select(-count)
print("Tabulation of variable values with censoring information:")
print(censored_tabulation)

# Add columns for plotting
y_tabulation <- y_tabulation %>%
 mutate(
  bin_start = as.numeric(sub("-.*$", "", bin)),
  bin_end = as.numeric(sub("^.*-", "", bin)),
  bin_center = (bin_start + bin_end) / 2
 )

# Create histogram
histogram_y <- ggplot(y_tabulation,
                aes(x = bin_center, y = censored_count)) +
 geom_col(fill = "steelblue", color = "black", alpha = 0.7,
        width= bin_width*0.9) +
 scale_x_continuous(breaks = bin_breaks, limits = c(min_y, max_y)) +
 labs(x = x_label,
     y = "Frequency") +
 theme_minimal() +
 theme(
  panel.grid.major = element_blank(),
  axis.title.x = element_text(margin = margin( t = 15, r = 0, b = 0, l = 0)),
  axis.text.x = element_text(angle = 45, hjust = 1)
 )

print(histogram_y)
```

```
  ggsave(
    filename = filename,
    plot = histogram_y,
    width = #DELETED FOR CLEARANCE
    height = #DELETED FOR CLEARANCE
    dpi = #DELETED FOR CLEARANCE

  )

  return(list(
    tabulation = y_tabulation,
    histogram = histogram_y
  ))
}



## ----overall_baseline----------------------------------------------------------------------------------------------------
  # Filter only observations in the primary analysis
  primary_analysis <- master %>%
    filter(In_Primary_analysis==1)

  ngrt_overall_score_baseline <- create_histogram(
      data = primary_analysis,
      variable = "Overall_reading_score",
      x_label = "NGRT overall reading score, baseline",
      bin_width = 20,
      filename =#DELETED FOR CLEARANCE
)
```

```
## ----overall_baseline_byarm----------------------------------------------------------------------------------------------------

 treatment <- primary_analysis %>%

  filter(treatment_allocation=="Treatment")


 ngrt_overall_score_baseline_treatment <- create_histogram(

    data = treatment,

    variable = "Overall_reading_score",

    x_label = "NGRT overall reading score, baseline, treatment group",

    bin_width = 20,

    filename = #DELETED FOR CLEARANCE

)


 control <- primary_analysis %>%

  filter(treatment_allocation=="Control")


 ngrt_overall_score_baseline_control <- create_histogram(

    data = control,

    variable = "Overall_reading_score",

    x_label = "NGRT overall reading score, baseline, control group",

    bin_width = 20,

    filename = #DELETED FOR CLEARANCE

)




## ----pc_baseline---------------------------------------------------------------------------------------------------------------

 # Filter only observations in the primary analysis
```

```
master <- master %>%
mutate(
 In_PC_analysis = if_else(
   !is.na(treatment_allocation) & !is.na(randomisation_batch) &
    !is.na(PC_score) & !is.na(PC_score_el) & !is.na(YearGroup) & !is.na(gender_admin) &
    !is.na(eal_admin) & !is.na(month_of_test), 1, 0)
)

PC_analysis <- master %>%
 filter(In_PC_analysis==1)

nrow(PC_analysis)

PC_score_baseline <- create_histogram(
   data = PC_analysis,
   variable = "PC_score",
   x_label = "NGRT Passage Comprehension score, baseline",
   bin_width = 20,
   filename = #DELETED FOR CLEARANCE
)




## ----pc_baseline_byarm----------------------------------------------------------------------------------------------------
treatment <- PC_analysis %>%
  filter(treatment_allocation=="Treatment")

PC_baseline_treatment <- create_histogram(
```

```
    data = treatment,
    variable = "PC_score",
    x_label = "NGRT Passage Comprehension score, baseline, treatment group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



 control <- PC_analysis %>%
  filter(treatment_allocation=="Control")

 PC_baseline_control <- create_histogram(
    data = control,
    variable = "PC_score",
    x_label = "NGRT Passage Comprehension score, baseline, control group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)




## ----sc_baseline----------------------------------------------------------------------------------------------------------
 # Filter only observations in the primary analysis
master <- master %>%
 mutate(
   In_SC_analysis = if_else(
    !is.na(treatment_allocation) & !is.na(randomisation_batch) &
     !is.na(SC_score) & !is.na(SC_score_el) & !is.na(YearGroup) & !is.na(gender_admin) &
     !is.na(eal_admin) & !is.na(month_of_test), 1, 0)
```

```
 )

 SC_analysis <- master %>%
  filter(In_SC_analysis==1)

 SC_score_baseline <- create_histogram(
    data = SC_analysis,
    variable = "SC_score",
    x_label = "NGRT Sentence Completion score, baseline",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



## ----sc_baseline_byarm----------------------------------------------------------------------------------------------------
 treatment <- SC_analysis %>%
  filter(treatment_allocation=="Treatment")

 SC_baseline_treatment <- create_histogram(
    data = treatment,
    variable = "SC_score",
    x_label = "NGRT Sentence Completion score, baseline, treatment group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)


 control <- SC_analysis %>%
```

```
  filter(treatment_allocation=="Control")


 PC_baseline_control <- create_histogram(
    data = control,
    variable = "SC_score",
    x_label = "NGRT Sentence Completion, baseline, control group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)




## ----overall_endline----------------------------------------------------------------------------------------------------
 # Filter only observations in the primary analysis
 ngrt_overall_score_endline <- create_histogram(
    data = primary_analysis,
    variable = "Overall_reading_score_el",
    x_label = "NGRT overall reading score, endline",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)




## ----overall_endline_byarm----------------------------------------------------------------------------------------------
 treatment <- primary_analysis %>%
  filter(treatment_allocation=="Treatment")
```

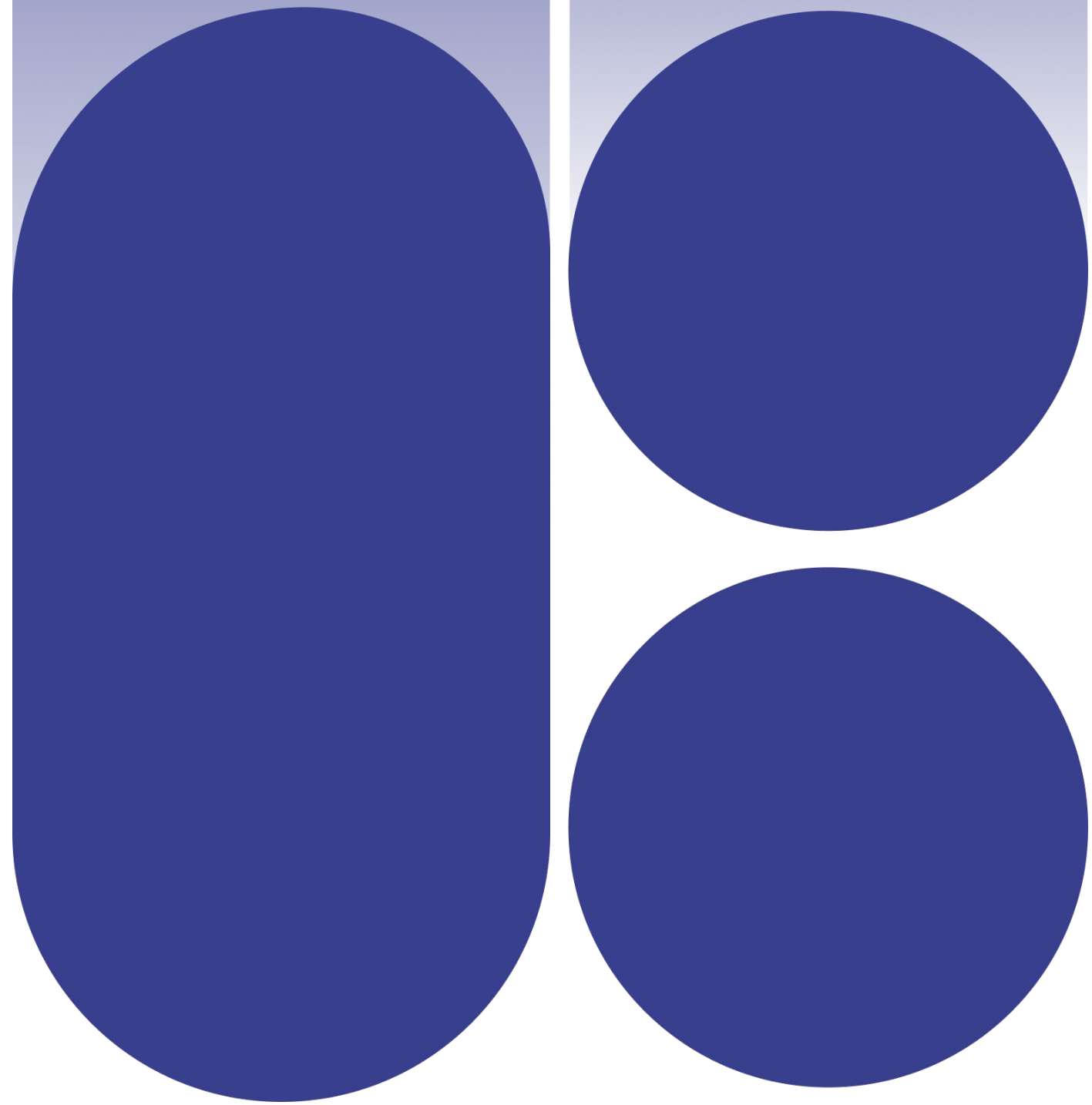
```
 ngrt_overall_score_endline_treatment <- create_histogram(
    data = treatment,
    variable = "Overall_reading_score_el",
    x_label = "NGRT overall reading score, endline, treatment group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



 control <- primary_analysis %>%
  filter(treatment_allocation=="Control")

 ngrt_overall_score_endline_control <- create_histogram(
    data = control,
    variable = "Overall_reading_score_el",
    x_label = "NGRT overall reading score, endline, control group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



## ----pc_endline----------------------------------------------------------------------------------------------------------
 # Filter only observations in the primary analysis

 PC_score_endline <- create_histogram(
    data = PC_analysis,
    variable = "PC_score_el",
    x_label = "NGRT Passage Comprehension score, endline",
```

```
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



## ----pc_endline_byarm----------------------------------------------------------------------------------------------------
 treatment <- PC_analysis %>%
  filter(treatment_allocation=="Treatment")

 PC_endline_treatment <- create_histogram(
    data = treatment,
    variable = "PC_score_el",
    x_label = "NGRT Passage Comprehension score, endline, treatment group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)



 control <- PC_analysis %>%
  filter(treatment_allocation=="Control")

 PC_endline_control <- create_histogram(
    data = control,
    variable = "PC_score_el",
    x_label = "NGRT Passage Comprehension score, endline, control group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)
```

```
## ----sc_endline----------------------------------------------------------------------------------------------------------

 # Filter only observations in the primary analysis


 SC_score_endline <- create_histogram(
    data = SC_analysis,
    variable = "SC_score_el",
    x_label = "NGRT Sentence Completion score, endline",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)




## ----sc_endline_byarm----------------------------------------------------------------------------------------------------
 treatment <- SC_analysis %>%
   filter(treatment_allocation=="Treatment")


 SC_endline_treatment <- create_histogram(
    data = treatment,
    variable = "SC_score_el",
    x_label = "NGRT Sentence Completion score, endline, treatment group",
    bin_width = 20,
    filename = #DELETED FOR CLEARANCE
)
```

```
 control <- SC_analysis %>%
  filter(treatment_allocation=="Control")

 PC_endline_control <- create_histogram(
   data = control,
   variable = "SC_score_el",
   x_label = "NGRT Sentence Completion, endline, control group",
   bin_width = 20,
   filename = #DELETED FOR CLEARANCE
)



## ----ks2----------------------------------------------------------------------------------------------------------------------
master <- master %>%
 mutate(
  In_ks2_analysis = if_else(
   !is.na(treatment_allocation) & !is.na(randomisation_batch) &
    !is.na(KS2_READSCORE_num) & !is.na(Overall_reading_score), 1, 0)
 )
 ks2_analysis <- master %>%
  filter(In_ks2_analysis==1)

 ks2_reading <- create_histogram(
   data = ks2_analysis,
   variable = "KS2_READSCORE_num",
   x_label = "KS2 Reading score",
```

```
    bin_width = 2,
    filename = #DELETED FOR CLEARANCE
)

## ----ks2_byarm----------------------------------------------------------------------------------------------------------
 treatment <- ks2_analysis %>%
  filter(treatment_allocation=="Treatment")

 ks2_treatment <- create_histogram(
    data = treatment,
    variable = "KS2_READSCORE_num",
    x_label = "KS2 Reading score, treatment group",
    bin_width = 2,
    filename = #DELETED FOR CLEARANCE
)
 control <- ks2_analysis %>%
  filter(treatment_allocation=="Control")

 ks2__control <- create_histogram(
    data = control,
    variable = "KS2_READSCORE_num",
    x_label = "KS2 Reading score, control group",
    bin_width = 2,
    filename = #DELETED FOR CLEARANCE
)
```



Education Endowment Foundation

The Education Endowment Foundation
5th Floor, Millbank Tower,
21–24 Millbank,
London,
SW1P 4QP

https://educationendowmentfoundation.org.uk

@EducEndowFoundn

Facebook.com/EducEndowFoundn